НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп'ютерних систем

До захисту допущено

Завідувач кафедри
_____ Віталій РОМАНКЕВИЧ
(підпис) (ініціали, прізвище)
"___" _____ 202___ р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Спеціалізовані комп'ютерні системи»

спеціальності **123 «Комп'ютерна інженерія»**

на тему: Модуль оброблення вхідного потоку даних для програмного бота

Виконав (-ла):

студент (-ка) IV курсу, групи КВ-62
2(шифр групи)

Ходоровський Андрій Петрович_____ _____
(прізвище, ім'я, по батькові) (підпис)

Керівник к.т.н., доцент Потапова К.Р. _____ _____
_____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант з нормоконтролю, доц.каф.СПСКС, к.т.н. Клятченко Я.М. _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент_____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020 року

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЄКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проєкт	2	
2	А4	ІАЛЦ.045490.001 ОА	Опис альбому	2	
3	А4	ІАЛЦ.045490.002 ТЗ	Технічне завдання	4	
4	А4	ІАЛЦ.045490.003 ТП	Відомість технічного проєкту	2	
5	А4	ІАЛЦ.045490.004 ПЗ	Пояснювальна записка	49	
6	А4	ІАЛЦ.045490.005 Д1	Міжмодульна взаємодія. Схема структурна	1	
7	А4	ІАЛЦ.045490.006 Д2	Схема взаємодії програми з даними. Схема алгоритму	1	
8	А4	ІАЛЦ.045490.007 Д3	Схема декодування зображення. Схема алгоритму	1	
9	А4	ІАЛЦ.045490.008 Д4	Схема управління периферією. Схема алгоритму	1	

<table>
<tr><td></td><td></td><td></td><td></td><td colspan="3">ІАЛЦ.045490.000</td></tr>
<tr><td></td><td>ПІБ</td><td>Підп.</td><td>Дата</td><td colspan="3"></td></tr>
<tr><td>Розробн.</td><td>Ходоровський А.П.</td><td></td><td></td><td rowspan="5">Відомість дипломного проєкту</td><td>Лист</td><td>Листів</td></tr>
<tr><td>Керівн.</td><td>Потапова К.Р.</td><td></td><td></td><td>1</td><td>1</td></tr>
<tr><td>Консульт.</td><td></td><td></td><td></td><td rowspan="3" colspan="2">КПІ ім. Ігоря Сікорського
Каф. СПіСКС
Гр. КВ-62</td></tr>
<tr><td>Н/контр.</td><td></td><td></td><td></td></tr>
<tr><td>Зав.каф.</td><td>Романкевич В.О.</td><td></td><td></td></tr>
</table>

Пояснювальна записка
до дипломного проєкту

на тему: Модуль оброблення вхідного потоку даних для програмного бота

Київ – 2020 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп'ютерних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність 123 «Комп'ютерна інженерія»

Освітньо-професійна програма «Спеціалізовані комп'ютерні системи»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Віталій РОМАНКЕВИЧ
(підпис) (ініціали, прізвище)
«___» _____20___ р.

ЗАВДАННЯ
на дипломний проєкт студента
Ходоровський Андрій Петрович
(прізвище, ім'я, по батькові)

1. Тема проєкту «модуль оброблення вхідного потоку даних для програмного бота»,
керівник проєкту Потапова Катерина Романівна, к.т.н, доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «25» травня 2020 р. № N1181-С.

2. Термін подання студентом проєкту

3. Вихідні дані до проєкту див. Технічне завдання

4. Зміст пояснювальної записки
- аналіз існуючих рішень;
- опис засобів для вирішення поставленої задачі;
- алгоритми обробки зображень;

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо)

- Міжмодульна взаємодія. Схема структурна
- Схема взаємодії програми з даними. Схема алгоритму
- Схема декодування зображення. Схема алгоритму
- Схема управління периферією. Схема алгоритму

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Клятченко Я.М. доцент		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
	Вивчення літератури за тематикою проєкту	10.02.2020	
	Узгодження технічного завдання	20.03.2020	
	Розробка програмного забезпечення	25.03.2020	
	Відлагодження програмного продукту	03.04.2020	
	Підготовка матеріалів першого розділу дипломного проєкту	11.04.2020	
	Підготовка матеріалів другого розділу дипломного проєкту	16.04.2020	
	Підготовка матеріалів третього розділу дипломного проєкту	28.04.2020	
	Підготовка графічної частини дипломного проєкту	08.05.2020	
	Оформлення документації дипломного проєкту	10.05.2020	

Студент _____ ХОДОРОВСЬКИЙ АНДРІЙ
(підпис)

Керівник проєкту _____ ПОТАПОВА КАТЕРИНА
(підпис)

АННОТАЦІЯ

Бакалаврська робота включає пояснювальну записку (49 ст., 38 рис., 1 табл.).

Об’єкт розробки – алгоритм обробки та використання зображень, та їх подальший аналіз розробленим програмним забезпеченням для взаємодії із користувацьким інтерфейсом.

Програмний модуль дозволяє знаходити за допомогою аналізу вихідного потоку даних периферійного пристрою, в даному випадку екрану, точки дотику, за якими було здійснено пошук. Результатом роботи є знаходження координат масиву через глобальні мінімуми або максимуми функції.

Додаток розроблено на базі мови програмування C++ за допомогою бібліотек: windows для обробки механічних рухів мишею, OpenCV для знаходження шаблонів та їхніх координат та GDI+ для управління графікою. В процесі користувалися середовищем розробки MS Visual Studio.

В ході розробки:

- проведено аналіз існуючих рішень для даної задачі;
- сформульовано вимоги для програмного забезпечення для дотримання високої продуктивності кінцевого продукту;
- проаналізовано за певними критеріями алгоритми пошуку співпадінь для конкретної задачі;
- розроблено програмне забезпечення на базі алгоритм аналізу.

Ключові слова:

OPENCV, АЛГОРИТМ CCOEFF_NORMED, C++, НОРМАЛІЗАЦІЯ, ШАБЛОН, MATCHTEMPLATE, КОМП’ЮТЕРНИЙ ЗІР, ОБРОБКА ЗОБРАЖЕНЬ

ANNOTATION

The thesis includes an explanatory note (49 pages, 38 images, 1 table).

The object of development - the algorithm for processing images, and further analyses of these images with the program for interaction with a user interface.

The program allows you to find by analyzing the output data stream of the peripheral device, in this case, the screen, the touchpoints which were searched. The result is finding the coordinates of the array through the global minima or maxima of the function.

The addition was made with language C++, using libraries: windows for processing mechanic mouse moves, OpenCV is used for searching templates and their coordinates, and GDI+ for graphic management. In process was used environment called Visual Studio.

During development:

- analyzed already done works for this task;
- formulated requirements for software to maintain high performance of the final product;
- algorithms for finding matches for a specific task are analyzed according to certain criteria;
- software based on analysis algorithm is developed.

Keywords: OPENCV, ALGORITHM CCOEFF_NORMED, C++, NORMALIZING, TEMPLATE, MATCHTEMPALTE, COMPUTER VISION, PICTURE PROCESSING

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.045490.002 ТЗ	Модуль оброблення	4		
			вхідного потоку даних			
			для програмного бота			
			Технічне завдання			
	А4	ІАЛЦ.045490.003 ТП	Модуль оброблення	2		
			вхідного потоку даних			
			для програмного бота			
			Відомість технічного			
			проекту			
	А4	ІАЛЦ.045490.004 ПЗ	Модуль оброблення	51		
			вхідного потоку даних			
			для програмного бота			
			Пояснювальна записка			
	А4	ІАЛЦ.045490.005 Д1	Міжмодульна взаємодія	1		
			Схема структурна			

					ІАЛЦ.045490.001 ОА
Змін.	Арк.	№ докум.	Підпис	Дата	
Розробив		Ходоровський АП			Модуль оброблення вхідного потоку даних для програмного бота **Опис альбому**
Перевірив		Потапова К.Р.			
Консульт.					
Н. контроль		Клятченко Я.М.			
Зав. каф.		Романкевич В.О			

Літ.	Аркуш	Аркушів
	1	2

КПІ ім. Ігоря Сікорського, ФПМ КВ-62

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.045490.006 Д2	Схема взаємодії програми	1		
			з даними			
			Схема алгоритму			
	А4	ІАЛЦ.045490.007 Д3	Схема декодування	1		
			зображення			
			Схема алгоритму			
	А4	ІАЛЦ.045490.008 Д4	Схема управління	1		
			периферією			
			Схема алгоритму			
		Диск CD-ROM	Текст пояснювальної	1		
			записки.			
			Графічний матеріал			

					ІАЛЦ.045490.001 ОА	Арк.
						2
Змін.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

					ІАЛЦ.045490.002 ТЗ		
Зм	*Лист*	*№ докум.*	*Підп.*	*Дата*			
Розроб.		Ходоровський			Модуль оброблення вхідного потоку даних для програмного бота	*Літ.*	*Лист* 1 / *Листів* 63
Перев.		Потапова					
Н. контр.		Клятченко			**Технічне завдання**	**НТУУ «КПІ ім. Ігоря Сікорського», ФПМ, КВ-62**	
Затв.		Романкевич					

НАЙМЕНУВАННЯ ТА ГАЛУЗЬ РОЗРОБКИ

Назва розробки: «Модуль оброблення вхідного потоку даних для програмного бота».

Галузь застосування: ігрова індустрія, сфера розваг.

ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на дипломне проектування на здобуття першого (бакалаврського) рівня вищої освіти, затверджене кафедрою системного програмування і спеціалізованих комп'ютерних систем Національного технічного університету України «Київський Політехнічний Інститут імені Ігоря Сікорського».

МЕТА І ПРИЗНАЧЕННЯ РОБОТИ

Метою даного проекту є створення алгоритму оброблення вхідного потоку даних для програмного бота.

ДЖЕРЕЛА РОЗРОБКИ

Джерелом інформації є технічна та науково-технічна література, технічна документація, публікації у періодичних виданнях та електронні статті у мережі Інтернет.

ТЕХНІЧНІ ВИМОГИ

Вимоги до програмного продукту, що розробляється

- сумісність з операційною системою Windows;
- аналіз отриманих зображень з екрану користувача;
- можливість збереження зображень на носії інформації;
- висока точність роботи алгоритму;

Вимоги до апаратного забезпечення

- Процесор: x86-сумісний 2,6 ГГц;
- Відеокарта: NVIDIA GeForce GTX 650 TI або ATI Radeon HD 7850;
- Оперативна пам’ять: 4 Гб;
- Місце на диску: 32 Гігабайта;
- Наявність доступу до мережі Internet (GPRS, EDGE, 3G, 4G);

Вимоги до програмного та апаратного забезпечення користувача

- Операційна система Windows 7 SP1;
- DirectX: версія 11;

ЕТАПИ РОЗРОБКИ

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів
1.	Вивчення літератури за тематикою проекту	10.02.2020
2.	Узгодження технічного завдання	20.03.2020
3.	Розробка програмного забезпечення	25.03.2020
4.	Відлагодження програмного продукту	10.04.2020
5.	Підготовка матеріалів першого розділу дипломного проекту	11.04.2020
6.	Підготовка матеріалів другого розділу дипломного проекту	16.04.2020
7.	Підготовка матеріалів третього розділу дипломного проекту	28.04.2020
8.	Підготовка графічної частини дипломного проекту	08.05.2020
9.	Оформлення документації дипломного проекту	10.05.2020
10.	Передзахист дипломного проекту	20.05.2020

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.045490.004 ПЗ	Модуль оброблення	49		
			вхідного потоку даних для			
			програмного бота			
			Пояснювальна записка			
	А4	ІАЛЦ.045490.005 Д1	Міжмодульна взаємодія	1		
			Схема структурна			
	А4	ІАЛЦ.045490.006 Д2	Схема взаємодії програми	1		
			з даними			
			Схема алгоритму			
	А4	ІАЛЦ.045490.007 Д3	Схема декодування	1		
			зображення			
			Схема алгоритму			
	А4	ІАЛЦ.045490.008 Д4	Схема управління	1		
			периферією			
			Схема алгоритму			

					ІАЛЦ.045490.003 ТП
Змін.	Арк.	№ докум.	Підпис	Дата	
Розробив		ХодоровськийА.П			Модуль оброблення вхідного потоку даних для програмного бота
Перевірив		Потапова К.Р.			
Консульт.					
Н. контроль		Клятченко Я.М.			
Зав. каф.		Романкевич В.О			

Літ.	Аркуш	Аркушів
	1	2

КПІ ім. Ігоря Сікорського, ФПМ КВ-62

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
		Диск CD-ROM	Текст пояснювальної	1		
			записки.			
			Графічний матеріал			

Змін.	Арк.	№ докум.	Підпис	Дата	ДП.45490.003 ТП	Арк. 2

ЗМІСТ

ВСТУП

Комп’ютерний зір – це інженерна галузь, яка вирішує питання автоматизації аналізу цифрового зображення. Це зображення, яке подається для аналізу, може бути представленим як: відео у записі, зображення, відео у реальному часі, дані для 3D-сканерів, принтерів. Найголовнішим завданням для технології комп’ютерного зору є обробка та аналіз цифрової інформації. Процес аналізу базується на перетворені цієї інформації у певні числа або символи, котрі може сприймати і аналізувати людина. Теоретичні розробки та моделі комп’ютерного зору використовуються для перетворення інформації із систем, котрі оброблюють цю інформацію. Багато методів пов’язаних з алгоритмами роботи беруть свій початок з різноманітних джерел. Таким чином, ця галузь обробки інформації пов’язана із: геометрією, статистикою, фізикою, нейробіологією, математичною оптимізацією, обробкою сигналів, квантовою механікою, штучним інтелектом, оптикою, фізикою різноманітних твердих тіл. Так, вивчення однієї біологічної системи можна пізніше дослідити за допомогою відтворення цієї ж системи, але вже з використанням штучних систем.

Комп’ютерний зір входить у такі системи обробки даних як: система для підрахунку кількості потрібних об’єктів, ідентифікації номерних знаків на автомобілях, розрізняння символів, розпізнавання людських лиць, особливостей чи деформацій об’єктів , їх виявлення, знаходження за різними критеріями пошуку всіх потрібних зображень серед каталогу зображень. Також, дана технологія може вирішити і питання відновлення зображень шляхом видалення шумів з зображення. Для цього використовуються різноманітні за складністю фільтри.

Важливим є застосування технології комп’ютерного зору у медицині. Прикладом використання є: виявлення різноманітних пухлин, деформацій у

скелеті, проблем із кровотоком. Технологію прибирання шумів використовують для покращення рентгенівських знімків чи ультразвукових зображень.

У військовій справі комп'ютерний зір використовується для виявлення ворожих сил та наведення на них ракет, отримання інформації із різних датчиків інформації, для зменшення складності обробки даних та для збільшення надійності отриманих результатів. Також використовують безпілотні літальні апарати (БПЛА) для розвідки.

Прикладом автономних транспортних засобів є безпілотники наземних транспортних засобів, стерео камери встановлені на гвинтокрилах. Рівень автономії поступово збільшується. Наразі існують повністю автономні транспортні засоби та транспортні засоби, у котрих системи комп'ютерного зору підтримують водія у різноманітних ситуаціях. В даному випадку, такі системи дають актуальну інформацію щодо місця розташування автомобіля, навігації. Найпростішим прикладом може бути функція автопаркування. На теперішній час безпілотні автомобілі все більше набувають розвитку і популярності.

Системи розуміння зображень мають 3 рівня складності, в залежності від того, що потрібно знайти на зображені:

1. Примітиви зображень;
2. Межі, поверхні , обсяги;
3. Об'єкти, сцени, події.

Типова система на базі комп'ютерного зору складається з таких частин:

1. Одержання потрібного зображення;
2. Первинна обробка. У цей етап може входити: підвищення контрастності, видалення шумів, приведення до певного розміру зображення (масштабування);
3. Відокремлення деталей (лінії, межі об'єктів, точки інтерфейсів об'єктів);
4. Сегментація (виокремлення важливих точок або ділянок);

5. Кінцева обробка. На цьому етапі від вхідного зображення може вже залишитись лише координати декількох точок, тому відбувається перевірка даних на задоволення умови, або їх класифікація.

Практично усі системи комп'ютерного зору містять основні елементи, потрібні для її функціонування:

- джерело живлення;
- елемент для зберігання зображень;
- процесор для виконання алгоритмів обробки;
- програмне забезпечення для обробки зображень.

Актуальність вибраної дипломної теми полягає у використанні технології комп'ютерного зору для забезпечення автоматизації пошуку, наведення та підбору предметів.

Об'єкт дослідження – екранна інформація, яку отримує користувач від гри.

Мета дипломного проекту: написання алгоритму обробки екранної інформації та програмного забезпечення для підбору предметів у грі Path of Exile.

Для досягнення поставленої мети дипломного проєкту було поставлено такі завдання:

- аналіз існуючих рішень;
- опис засобів для вирішення поставленої задачі;
- аналіз алгоритмів обробки зображення;
- написання програмного забезпечення;
- висновки з результатів роботи програмного модуля.

1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

Із доступних виділяється програмне забезпечення від thebbandit - “[WingmanReloaded] Auto (Flask, Quit, Mine, Spell) & Loot (Manage, Filter, Click)”. Дане програмне забезпечення має обширну дію і впливає не лише на підбір.

Автоматизація натискань колб та заклинань.

Дане програмне забезпечення може автоматизувати повторювання певних завдань, котрі ви вказуєте. Також, присутній самостійний вихід при низькому рівні здоров’я, що дозволяє гравцеві не втратити цінний досвід. Присутня можливість налаштування відтворення до 5 заклинань та автоматизація підривання мін. Також, можливо налаштувати натискання певних клавіш при використанні певної колби. Вигляд даного програмного забезпечення на рис.1.1.

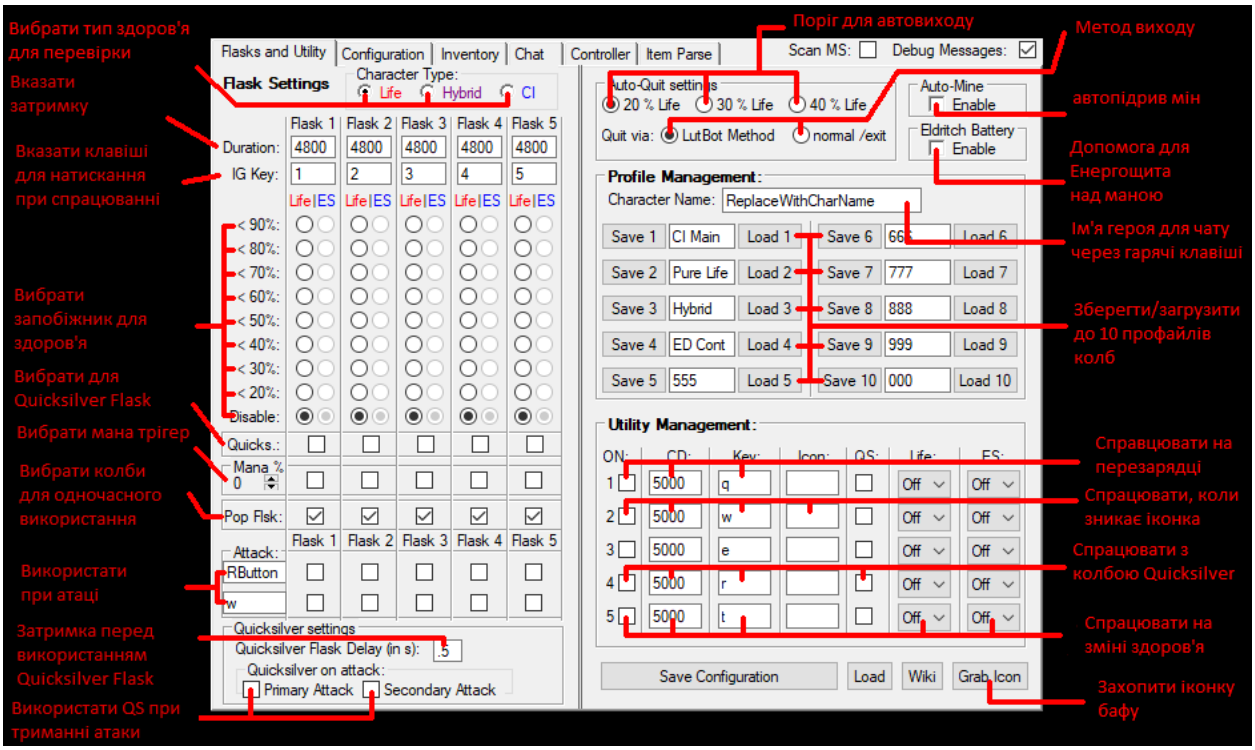

Рисунок 1.1 – інтерфейс програми

Одна кнопка для управління всіма функціями інвентаризації.

Програмне забезпечення самостійно сортує ваш інвентар, ідентифікує елементи та надсилає їх на призначену вкладку сховища або продає постачальнику-ніпу. Також, можливим є обмін повністю зібраних пакунків карт ворожіння рис.1.2.

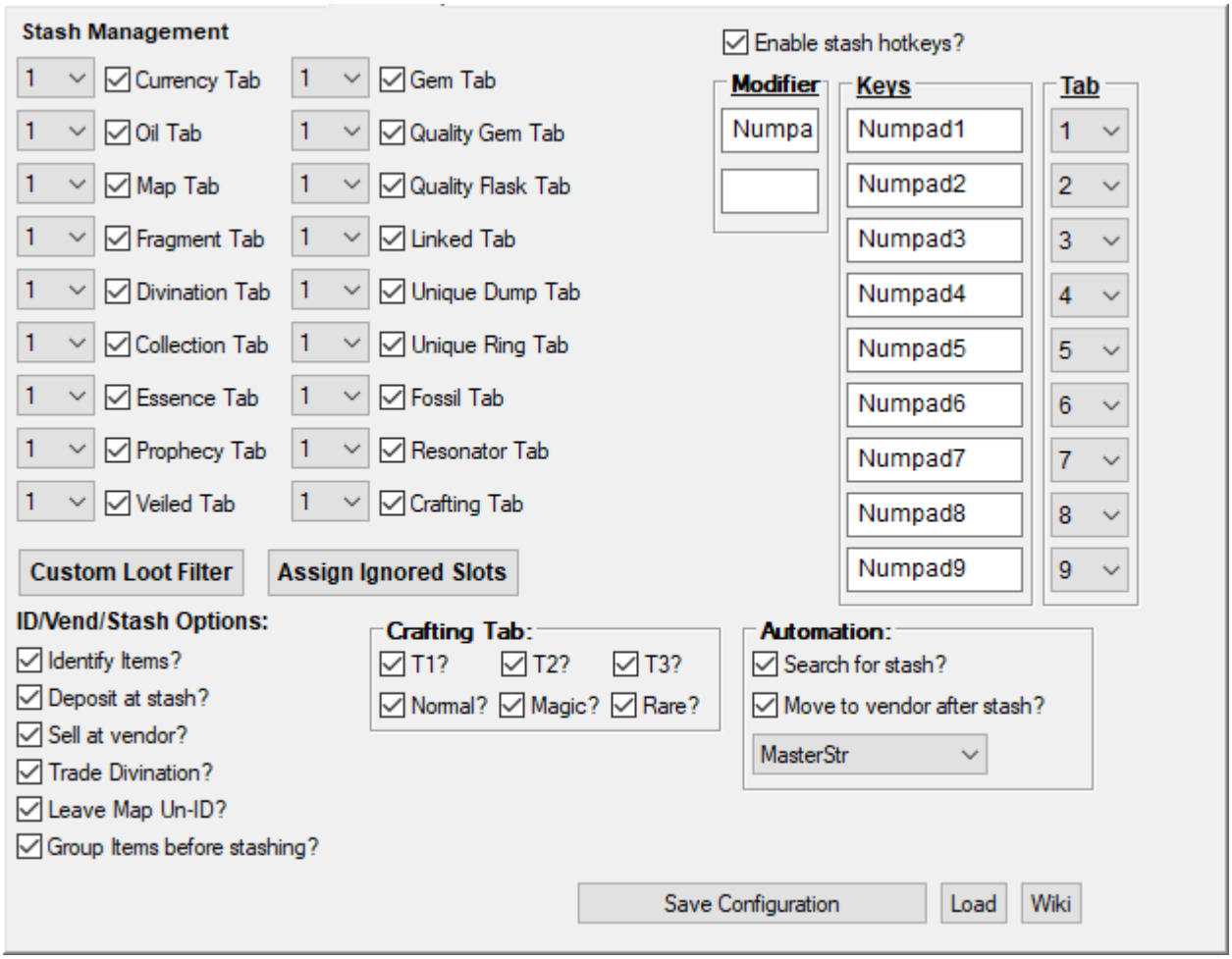

Рисунок 1.2 – взаємодія з вкладками користувача

Робота з підбором предметів на рис.1.3.

У цьому випадку, кнопка підбору починає пошук предметів. Якщо програмне забезпечення знаходить, то наступною дією буде спроба натискання. Також, можна натиснути на кнопку підбору та наводити на імена предметів – це спричинить підбір.

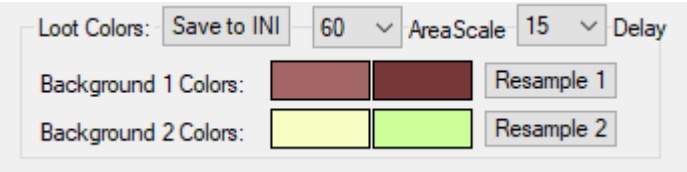

Рисунок 1.3 – інтерфейс для роботи із підбором

Спеціальні фільтри рис.1.4.

Налаштування фільтрів для предметів, які оцінюватимуть статистику, префікси, суфікси, імпліцити.

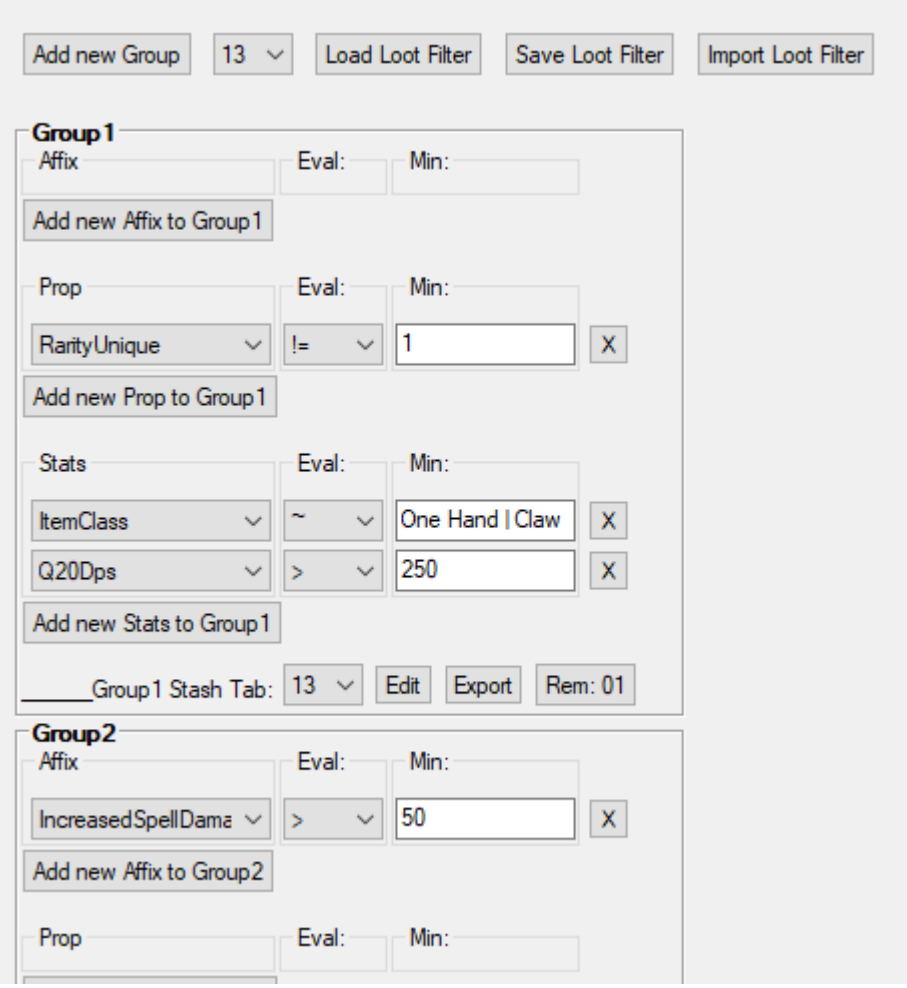

Рисунок 1.4 – інтерфейс для оцінювання речей

Чат команди та гарячі клавіші для відповіді рис.1.5.

Налаштування до 9 гарячих клавіш для управління чат командами. Надсилання запрошень до груп, котрі щойно вам написали, або дайте знати їм,

що ви скоро повернетеся. Також, можна швидко перейти до свого сховища, межанерії чи очистити чат.

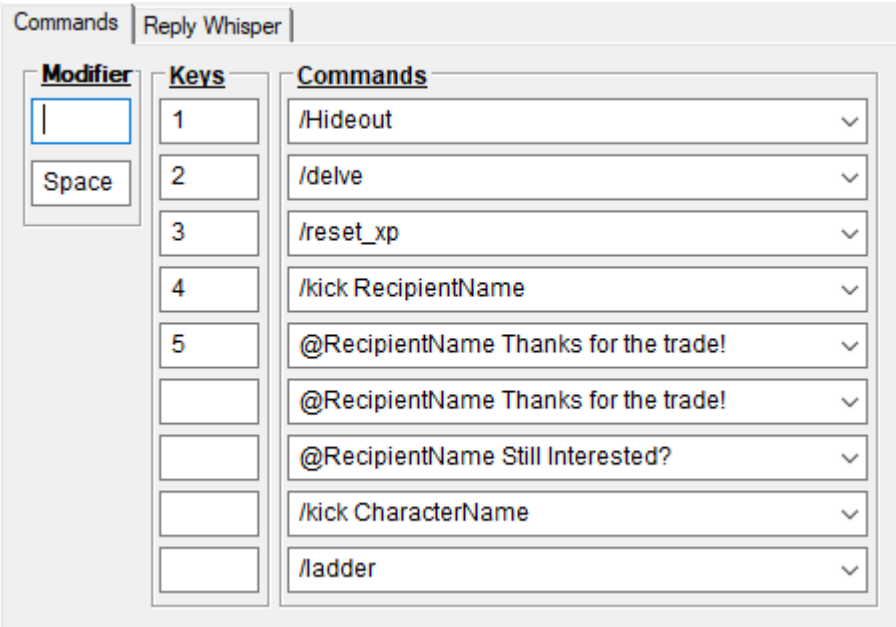

Рисунок 1.5 – інтерфейс призначення клавіш для користувача

Підтримка роботи з телевізора рис 1.6.

Увімкніть підтримку контролера для ігрового потоку персонального комп’ютера та грайте з дивана. Більшість функцій з програми можна прив’язати до контролера.

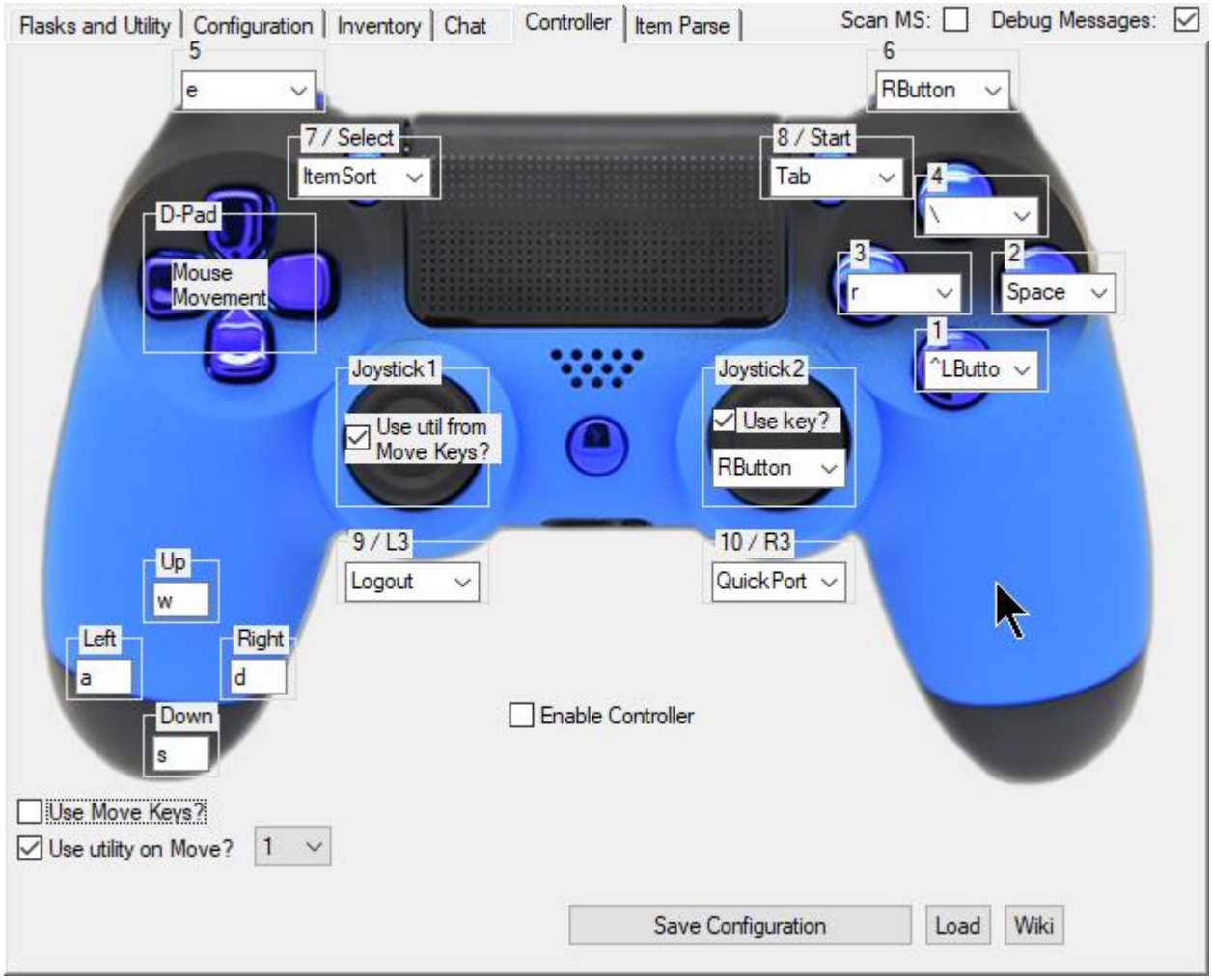

Рисунок 1.6 – можливе призначення клавіш для джойстика

Програмне забезпечення має можливість самостійно перевіряти наявність оновлень даного програмного забезпечення. При наявності буде задано питання чи потрібно його встановлювати.

Вимоги, котрі ставить дане програмне забезпечення перед користувачем:

- повноекранний режим екрану;
- підтримка співвідношень сторін;
- стандартне співвідношення аспектів - 16:9. Загальна роздільна здатність – 1920 x 1080;
- класичне співвідношення аспектів – 12:9 (4:3). Загальна роздільна здатність – 1440 x 1080 (800 x 600);
- кінематичне співвідношення сторін – 21:9. Загальна роздільна здатність – 2560 x 1080.;

- широкоформатне співвідношення – 32:9. Загальна роздільна здатність – 3840 x 1080;

Офіційний клієнт пропонує автоматизований підбір предметів для клієнтів з Азійського регіону. Дана риса є безкомпромісно найкращим варіантом, оскільки все відбувається без встановлення стороннього програмного забезпечення і працює з «упаковки».

2. ОПИС ЗАСОБІВ ДЛЯ ВИРІШЕННЯ ПОСТАВЛЕНОЇ ЗАДАЧІ

2.1 C++

В основі програми лягла мова програмування C++. C++ це статична, типізована, компільована мова загального призначення, розроблена 1983р. Б’ярном Страуструпом. В мові підтримується декілька парадигм програмування:

- об’єктно-орієнтоване програмування;
- процедурне програмування;
- узагальнене програмування.

C++ є однією із найпопулярніших мов програмування. Її використовують для використання у розробці операційних систем, прикладних програм різного призначення, драйверів, високоефективних серверів та комп’ютерних ігор.

В перспективах мови є:

- внесення доповнень із бібліотеки boost у стандартну бібліотеку;
- внесення великих глобальних змін замість великої кількості незначних правок;
- зміни не повинні зменшувати вже досягнуту ефективність роботи програм;
- базовими напрямками розробки є доопрацювання засобів загального програмування, стандартизація механізмів паралельного програмування, доопрацювання механізмів безпечного програмування;
- в майбутньому не виключено, що буде добавлено різні механізми із інших мов програмування (наприклад автоматична збірка сміття).

У стандарті C++11 добавили такі речі:

- об’явлення константних виразів та константних функцій через contexpr;
- універсальну ініціалізацію;
- оператори присвоювання і конструктори із переносом на новий адресний рядок;
- вивід типів, а для шаблонів новий механізм auto і decltype;

- цикл по колекціям. Іншими словами, це цикл, який працює для кожного елемента певної колекції. Вказується змінна, котра на кожному повторі циклу буде уособлювати у собі значення елемента даної колекції;
- лямбда-вираз. Це безіменні функції, визначені в місці, де вони зразу ж і використовуються;
- добавлено нові необов'язкові модифікатори для опису віртуальних класів. Override – добавляє перевірку у батьківському класі на наявність співпадінь сигнатур батьківського класу з класом нащадком. Final – забороняє заміну даного методу іншими;
- добавлено можливість опису варіативних шаблонів. Варіативність наявна у невідомій кількості його аргументів;
- визначено ключове слово для константи нульового вказівника – nullptr;
- змінено семантику та частково синтаксис перечислень та об'єднань;
- добавлено можливість створювати безпечні перерахування;
- з об'єднань знято заборону на структуру;

Описано лише в загальному стандарт 11 року, але потрібно розуміти, що з кожним новим стандартом (C++14, C++17) добавляються та змінюються речі, при реалізації яких могли виникати питання чи проблеми. Новий стандарт це не лише добавлення чи змінення старого функціоналу, у нових стандартах часто добавляють і новий функціонал: нові функції, нові операції для вже існуючих типів даних, нове представлення чи перетворення одних даних у інші. Потрібно розуміти також чому стандарти виходять через такий період часу. Кожен новий стандарт змушує програмістів не лише знаходити проблеми у реалізації того чи іншого програмного забезпечення, але і виправляти їх. Зрозуміло, що C++ мова достатньо швидка, тому оптимізацію уже існуючих рішень також потрібно виконувати для навіть незначних прискорень. Відлагодження програми може займати багато часу, якщо мова, на якій вона написана, буде недостатньо зрозуміла, або викликатиме багато запитань у програміста.

Потрібно розуміти, що у теперішній час мало хто працює в команді із однієї людини, написання доступного для швидкого сприйняття та розуміння коду є не менш важливим етапом у розробці програмного забезпечення у команді.

Існують певні шаблони написання коду, яких дотримуються розробники. Ці шаблони можуть відноситися до певної мови чи компанії, в середині якої працюють програмісти під одним тегом.

Найбільш відомим і поширеним є використання певних правил під час написання складних слів. Найбільш відомими представниками, таких правил, є: верблюжий регістр (CamelCase) та зміїний регістр (snake_case). Характерною рисою верблюжого регістра є те, що декілька слів пишуться разом без пробілів, але при цьому кожне нове слово починається з великої літери. Для іншого стилю характерною рисою є розділення кожного слова символом підкреслення “_“.

Верблюжий регістр використовують у таких мовах програмування:

- Java;
- .NET;
- Python (для назв класів);
- Wolfram.

Зміїний регістр використовуються у таких мовах:

- Perl;
- Python (для змінних, функцій та методів);
- Ruby (методи та змінні);
- Rust (для змінних функцій та модулів).

Вважається, що інформація написана зміїним регістром сприймається швидше ніж верблюжим.

Існує велика кількість мов програмування: від низькорівневих до високо, від мов, які спеціалізуються на вирішенні задач у певній сфері і так далі. Потрібно зазначити переваги цієї мови над іншими:

- містить засоби контролю ефективності розробки;
- висока сумісність C++ із C;

- продуктивність обчислювань;
- підтримка різних парадигм програмування: структурної, об’єктно-орієнтованої, функціональної, узагальненої та метапрограмування;
- автоматичний виклик деструкторів об’єктів. За порядком FILO (First in last out) - перший зайшов, останнім вийшов;
- перевантаження операторів;
- шаблони;
- можливо добавляти код із інших предметно-орієнтованих мов програмування;
- доступність. Для C++ існує дуже багато різноманітної літератури, перекладеної на велику кількість мов;
- має найбільш широкі можливості серед складних мов програмування.

Також зазначимо недоліки C++ перед іншими мовами програмування:

- відсутність системи модулів;
- наявність декількох механізмів вирішення одних і тих ж завдань (необережність може призвести до виникнення помилок, неоптимальних рішень);
- адресна арифметика, неявне приведення типів, можливість управління пам’яттю;
- використання шаблонів може призвести до написання неефективного і об’ємного коду;
- віртуальне спадкування;
- важкий синтаксис;
- об’ємна специфікація;
- складна для вивчення.

Мова C++ вплинула на розробку інших мов програмування, які в свою чергу виходили пізніше. Деякі мови програмування створювались для того, щоб вирішити проблеми, котрі виникали при розробці програмного

забезпечення на C++. Типовими конкурентами для C++ є : Objective-C, Java, Nemerle, F#, Go, Rust.

2.2 OpenCV

Це бібліотека комп'ютерного зору, обробки зображень з відкритим вихідним кодом для всіх. Реалізована на базі мов програмування C та C++, існують реалізації для таких мов програмування: Python, Java, C, C++, Ruby, Matlab, Lua і тд.

Область застосування даної бібліотеки є обширною. Перечислимо декілька важливих аспектів, в котрих використовується комп'ютерний зір:

- набори інструментів для роботи із 2D та 3D графікою;
- система розпізнавання обличчя;
- розпізнавання жестів;
- взаємодія між людиною та комп'ютером;
- робототехніка;
- розпізнавання рухів;
- ідентифікація об'єкта;
- сегментація;
- розпізнавання;
- сприйняття протяжності простору та глибини зображення від 2 камер;
- структура руху;
- відстеження руху;
- доповнена реальність.

OpenCV містить бібліотеку для статистичного машинного навчання, яка підтримує деякі з перерахованих вище областей. Дана бібліотека містить:

- градієнт для підсилення дерев;
- алгоритм очікування максимізації;
- k-алгоритм найближчого сусіда;
- наївний байєсів класифікатор;
- штучні нейронні мережі;

- випадкове створення лісу;
- підтримка векторної машини;
- глибокі нейронні мережі;
- прискорення;
- навчання на дереві рішень.

OpenCV підтримується на великій кількості настільних операційних системах наприклад: Windows, Linux, macOS, FreeBSD, NetBSD, OpenBSD. Та на мобільних: Android, IOS, Maemo, BlackBerry 10. Дану бібліотеку можна отримати із офіційних джерел на сайтах SourceForge, GitHub. OpenCV входить у основний пакет у ROS (операційна система робот). Також, існують такі ж програмні засоби як CVIPtools та OpenNN, які мають схожий нахил діяльності.

2.3 CONIO.H (CONSOLE INPUT-OUTPUT)

Даний заголовковий файл використовується для підключення функції _kbnit (оригінальна функція має прототип у вигляді int kbnit(void), але у Visual Studio 2019 цей прототип замінено на int _kbnit(void)), котра в свою чергу визначає чи була нажата клавіша користувачем.

Дане підключення використовувалось давніше у операційних системах MS-DOS. Вона не входить ні в один із стандартних наборів мов C або C++.

2.4 WINDOWS.H

Це спеціальний хедер файл, в котрому містять об’явлення всіх функцій пов’язаних із Windows API, усіх типів даних, усіх макросів. Також, даний хедер можна підключати і у мові C. Покажемо декілька дочірніх хедер файлів, котрі включені в склад windows.h:

- excpt.h – для обробки винятків;
- windef.h – макроси і типи;
- ctype.h – функції для класифікації символів, хедер відноситься до стандартних у мові C;
- string.h – рядки та буфери, хедер відноситься до стандартних у мові C;

- wingdi.h – складає користувацький інтерфейс та використовується для фіксування користувацького екрану.

2.5 matchTemplate

В ядрі програми лежить функція бібліотеки OpenCV – matchTemplate. Наведемо приклади прототипів даної команди на різних мовах:

1. На C++ прототип матиме такий вигляд: void matchTemplate (InputArray image, InputArray templ, OutputArray result, int method, InputArray mask=noArray());
2. На Python: result = cv2.matchTempalate(image, templ, method[, result[, mask]];
3. C: void cvMatchTemplate (const CvArr* image, const CvArr* templ, CvArr* result, int method).

Параметрам, вказаним у скобках, відповідають такі значення:

- image – зображення, у якому відбувається пошук (8 бітне або 32 з плаваючою комою);
- templ – шуканий шаблон. Він повинен співпадати за типом даних із типом зображення, у котрому відбувається пошук, Другою умовою буде те, що шаблон повинен бути меншим за початкове зображення;
- result – карта результату порівнянь. Вона одноканальна 32 бітна з плаваючою комою. Якщо вхідна картинка має параметри W на H, а шаблон w на h, то результат матиме такі параметри – (W – w – 1) на (H – h -1);
- method – параметр, котрий вказує використаний метод для порівняння.

Особливість порівняння - це аналіз особливостей на вхідному зображенні. До них відносяться: форми, текстури, кольори. Таку глибоку обробку можуть зробити нейронні мережі. На кожному етапі обробки нейронна мережа витягує якусь частину інформації з вхідного зображення, оброблює її і передає далі. Наступна ж нейронна мережа вже по-іншому оброблює отриману інформацію.

Вилучені особливості зображення є його умовною характеристикою. На рис.2.1. зображено перехід інформації від однієї ітерації до іншої.

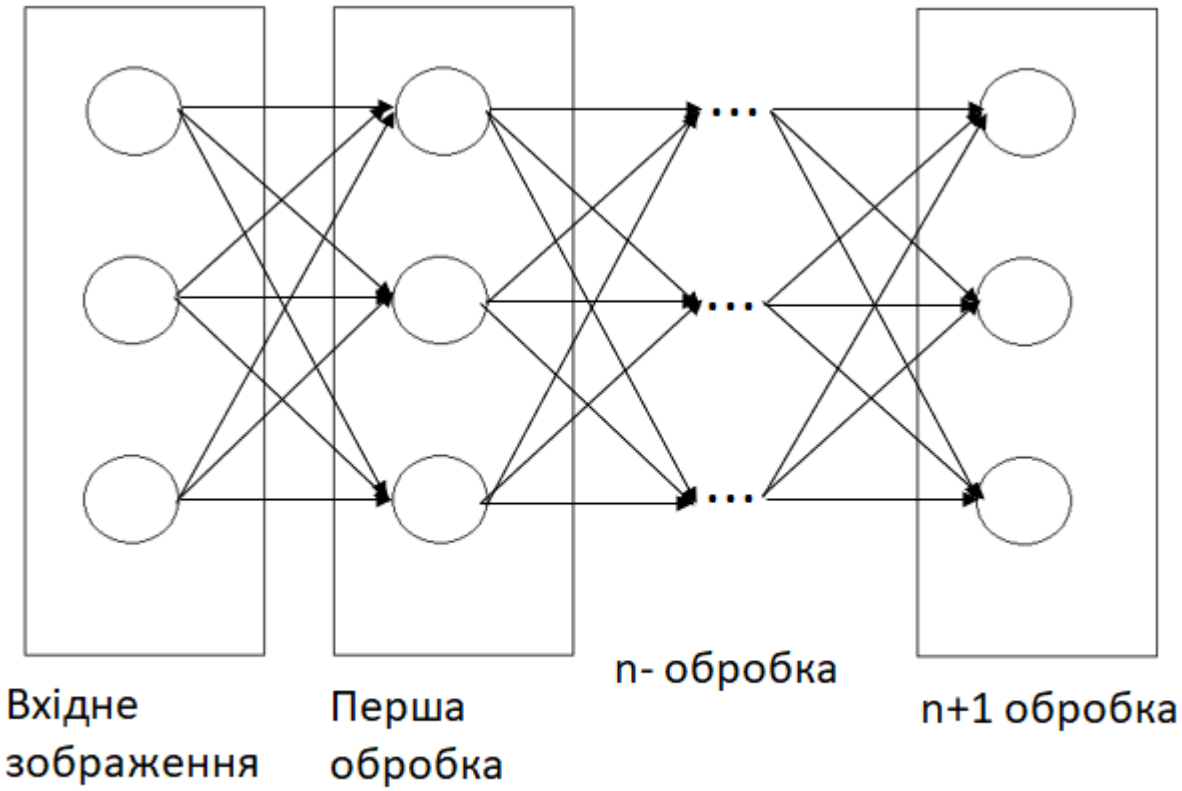

Рисунок 2.1 – робота алгоритму пошуку за шаблоном

Відповідність на базі якогось шаблону може вимагати перевірки великої кількості точок на співпадіння, за умови, якщо шаблон є великим чи зображення, тому, можна зменшити кількість перевірок, зменшивши кількість точок, тобто, знизити початкову роздільну здатність зображень.

Якщо шаблон на пряму не збігається, то є можливим використання шаблонів-просторів. Тобто, шаблонів, котрі по змісту є ідентичними до початкового. Різниця може бути в освітленості, точки зору, контрастності і так далі.

2.6 Коефіцієнт кореляції

Коефіцієнт кореляції - це числова міра деякого типу кореляції, що означає статистичну залежність між двома змінними. Змінні можуть бути двома базами

даних даного набору даних спостережень, які часто називають вибіркою.

Існує кілька типів коефіцієнта кореляції, кожен зі своїм визначенням та власним діапазоном зручності використання та характеристиками. Всі вони припускають значення в діапазоні від -1 до +1, де ± 1 вказує на найсильнішу можливу згоду, а 0 - на найсильнішу можливу незгоду. Як інструменти аналізу, коефіцієнти кореляції представляють певні проблеми, включаючи схильність деяких типів до спотворення та можливість їх неправильного використання для встановлення причинно-наслідкового зв'язку між змінними. Існує кілька різних можливостей для виміру ступеня кореляції даних, залежно від виду даних: вимірювальні, порядкові чи категоричні.

Коефіцієнт кореляції Пірсона - це міра сили та напряму лінійної залежності між двома змінними, що визначається, як коваріація змінних, поділених на добуток їхнього стандарту відхилення. Це найвідоміший і найчастіше використовуваний тип коефіцієнта кореляції. Коли термін "коефіцієнт кореляції" використовується без додаткової кваліфікації, він зазвичай відноситься до коефіцієнта кореляції Пірсона.

2.7 Перехресна кореляція

При обробці сигналів перехресна кореляція - це міра подібності двох рядів як функції переміщення одного відносно іншого. Це також відоме як розсувний крапковий виріб або розсувний внутрішній виріб. Він зазвичай використовується для пошуку довгого сигналу для більш короткої, відомої функції. Він має застосування в розпізнаванні зображень, аналізі одиничних частинок, електронної томографії, усередненні, криптоаналізі та нейрофізіології. Перехресна кореляція за своєю суттю схожа на згортку двох функцій. В автокореляції, яка є перехресною кореляцією сигналу з самим собою, завжди буде пік при відставанні до нуля, а його розмір буде енергією сигналу.

У вірогідності та статистиці термін перехресні кореляції відноситься до кореляцій між записами двох випадкових векторів X та У, тоді як кореляції випадкового вектора X є кореляціями між елементами цього вектора X. Це

формує кореляційну матрицю вектора X. Якщо X та У є скалярною випадковою змінною, яка реалізується неодноразово у часовому ряді, то кореляція різних тимчасових екземплярів вектора X називається аутокореляція X, а перехресні кореляції X з У протягом часу є тимчасовими перехресними кореляціями. У імовірності та статистиці визначення кореляції завжди включає коефіцієнт стандартизації таким чином, що кореляції мають значення між −1 та +1. Якщо X та У - дві незалежні випадкові величини з функціями густини ймовірностей f та g відповідно, то щільність ймовірності різниці У – X формально задається перехресною кореляцією $f \star g$.

Перехресна кореляція детермінованих сигналів. Для безперервних функцій f та g перехресна кореляція визначається за формулою

$$(f \star g)(\tau) \triangleq \int_{-\infty}^{\infty} \overline{f(t)} g(t+\tau) dt. \quad (1)$$

Це також еквівалентно

$$(f \star g)(\tau) \triangleq \int_{-\infty}^{\infty} \overline{f(t-\tau)} g(t) dt. \quad (2)$$

де $\overline{f(t)}$ означає складний сполучник $f(t)$.

τ це переміщення, також відоме як відставання (функція при f в t виникає в g при $t+\tau$).

Якщо f та g обидві безперервні періодичні функції періоду T інтеграція від $-\infty$ до ∞ замінюється інтеграцією через будь-який інтервал $[t_0, t_0+T]$ довжиною T:

$$(f \star g)(\tau) \triangleq \int_{t_0}^{t_0+T} \overline{f(t)} g(t+\tau) dt. \quad (3)$$

Це також еквівалентно

$$(f \star g)(\tau) \triangleq \int_{t_0}^{t_0+T} \overline{f(t-\tau)} g(t) dt. \quad (4)$$

Аналогічно, для дискретних функцій перехресна кореляція визначається як:

$$(f \star g)[n] \triangleq \sum_{m=-\infty}^{\infty} \overline{f[m]} g[m+n]. \quad (5)$$

Це також еквівалентно

$$(f \star g)[n] \triangleq \sum_{m=-\infty}^{\infty} \overline{f[m-n]} g[m]. \quad (6)$$

Для кінцевих дискретних функцій $f, g \in \mathbb{C}^N$, визначається (кругова) перехресна кореляція. як:

$$(f \star g)[n] \triangleq \sum_{m=0}^{N-1} \overline{f[m]} g[(m+n)_{\bmod N}]. \quad (7)$$

Це також еквівалентно

$$(f \star g)[n] \triangleq \sum_{m=0}^{N-1} \overline{f[(m-n)_{\bmod N}]} g[m]. \quad (8)$$

Для кінцевих дискретних функцій $f \in \mathbb{C}^N$, $g \in \mathbb{C}^N$, перехресна кореляція ядра визначається як:

$$(f \star g)[n] \triangleq \sum_{m=0}^{N-1} \overline{f[m]}\, K_g[(m+n)_{\bmod N}]\,. \quad (9)$$

де $K_g = [k(g, T_0(g)), k(g, T_1(g)), \ldots, k(g, T_{N-1}(g))]$ - вектор функцій ядра.

Як приклад, розглянемо дві функції f та g, що відрізняються лише невідомим зміщенням по осі x. Можна використовувати перехресну кореляцію, щоб знайти, на скільки g повинно бути зміщено вздовж осі x, щоб зробити його

ідентичним f. Формула по суті ковзає функцію g вздовж осі x, обчислюючи інтеграл їх добутку в кожній позиції. Коли функції збігаються, значення $(f * g)$ є максимальним. Це тому що, коли вирівнюються піки (позитивні зони), вони вносять великий внесок у інтеграл. Аналогічно, коли негативні зони вирівнюються, вони також роблять позитивний внесок у інтеграл, оскільки добуток двох від'ємних чисел є позитивним.

Перехресне співвідношення функцій $f(t)$ та $g(t)$ еквівалентно згортці (позначається як $*$) $\overline{f(-t)}$ і $g(t)$. Це є:

$$[f(t) \star g(t)](t) = [\overline{f(-t)} * g(t)](t)\,. \tag{10}$$

$$[f(t) \star g(t)](t) = [\overline{g(t)} \star \overline{f(t)}](-t)\,. \tag{11}$$

Якщо f це Ермітова функція, то

$$f \star g = f * g. \tag{12}$$

Якщо обидві f та g є Ермітовими функціями, то

$$f \star g = g \star f. \tag{13}$$

$$(f \star g) \star (f \star g) = (f \star f) \star (g \star g)\,. \tag{14}$$

Аналогічно теоремі згортки, крос-кореляція задовольняє

$$\mathcal{F}\{f \star g\} = \overline{\mathcal{F}\{f\}} \cdot \mathcal{F}\{g\}. \tag{15}$$

де $\mathcal{F}$ позначає перетворення Фур'є, і $\overline{f}$ знову вказує складний кон'югат f, через те що

$$\mathcal{F}\left\{\overline{f(-t)}\right\} = \overline{\mathcal{F}\{f(-t)\}}. \quad (16)$$

У поєднанні з алгоритмами швидкого перетворення Фур'є, ця властивість часто використовується для ефективного чисельного обчислення перехресних кореляцій.

Перехресне співвідношення згортки f і h з функцією g це згортання перехресного співвідношення g і f з ядром h:

$$g \star (f \star h) = (g \star f) \star h. \quad (17)$$

Перехресна кореляція стохастичних процесів. У аналізі часових рядів та статистиці перехресна кореляція пари випадкових процесів є співвідношенням між значеннями процесів у різний час, як функція двох разів. Нехай (X_t, Y_t) пара випадкових чисел, коли t - будь-який момент часу. Час може бути цілим числом для процесу дискретного часу або реальним числом для процесу безперервного часу. Тоді, X_t - це значення, отримане заданим запуском процесу в момент часу t. Припустимо, що процес має такі значення $\mu_X(t)$ і $\mu_Y(t)$ і дисперсії $\sigma_X^2(t)$ і $\sigma_Y^2(t)$ в час t. Тоді, визначення перехресної кореляції між часом t_1 і t_2 виглядатиме:

$$R_{XY}(t_1, t_2) = E[X_{t_1}\overline{Y_{t_2}}]\,. \quad (18)$$

де E - - оператор математичного сподівання.

Перехресна коваріаційна функція. Віднімання середнього значення перед множенням дає перехресну коваріацію між часом t_1, t_2. Виразом не можна чітко визначити для періодичних процесів, оскільки середнє значення може не існувати або дисперсія.

$$K_{XY}(t_1, t_2) = E[(X_{t_1} - \mu_X(t_1))\overline{(Y_{t_2} - \mu_Y(t_2))}]\,. \quad (19)$$

Визначення для широкого сенсу стаціонарного стохастичного процесу. (X_t, Y_t) - являють собою пару стохастичних процесів, які є спільно широкосмисленними нерухомими. Тоді функція перехресної коваріації та перехресна кореляція задаються наступним чином.

Перехресна кореляційна функція

$$R_{XY}(\tau) = E[X_t \overline{Y_{t+\tau}}] . \quad (21)$$

Це також еквівалентно

$$R_{XY}(\tau) = E[X_{t-\tau} \overline{Y_t}] . \quad (22)$$

Перехресна коваріаційна функція

$$K_{XY}(\tau) = E[(X_t - \mu_X)\overline{(Y_{t+\tau} - \mu_Y)}] . \quad (23)$$

Це також еквівалентно

$$K_{XY}(\tau) = E[(X_{t-\tau} - \mu_X)\overline{(Y_t - \mu_Y)}] . \quad (24)$$

де μ_X та σ_X - середнє та стандартне відхилення процесу (X_t), які постійні в часі через стаціонарність; і аналогічно для (Y_t), відповідно.

$E[]$ вказує на очікуване значення. Від чого перехресна коваріація та перехресна кореляція не залежать від t.

Перехресне співвідношення пари спільно широких смислових стаціонарних стохастичних процесів можна оцінити шляхом усереднення добутку зразків, виміряних від одного процесу, та зразків, виміряних від іншого (та його часових зрушень). Вибірки, що входять до середнього значення, можуть бути довільним підмножиною всіх вибірок сигналу (наприклад, зразки

в межах обмеженого часового вікна або під вибірки одного з сигналів). Для великої кількості вибірок середнє значення сходить до справжньої перехресної кореляції.

2.8 Нормалізація

Загальна практика в деяких дисциплінах (наприклад, статистика та аналіз часових рядів) нормалізує функцію перехресної кореляції, щоб отримати залежний від часу коефіцієнт кореляції Пірсона. Однак, в інших дисциплінах (наприклад, інженерії) нормалізація зазвичай відпадає, а терміни "перехресна кореляція" та "перехресна коваріація" використовуються взаємозамінно. Визначення нормованої перехресної кореляції стохастичного процесу є

$$\rho_{XX}(t_1, t_2) = \frac{K_{XX}(t_1,t_2)}{\sigma_X(t_1)\sigma_X(t_2)} = \frac{E[(X_{t_1}-\mu_{t_1})\overline{(Y_{t_2}-\mu_{t_2})}]}{\sigma_X(t_1)\sigma_X(t_2)}. \quad (25)$$

Якщо функція ρ_{XX} є чітко визначеною, її значення повинно лежати в діапазоні [-1,1], при цьому 1 вказує на ідеальну кореляцію, а -1 - на ідеальну антикореляцію.

Для спільно широко значущих стаціонарних стохастичних процесів це визначення набуває вигляду:

$$\rho_{XY}(\tau) = \frac{K_{XY}(\tau)}{\sigma_X\sigma_Y} = \frac{E[(X_t-\mu_X)\overline{(Y_{t+\tau}-\mu_Y)}]}{\sigma_X\sigma_Y}. \quad (26)$$

Нормалізація важлива і тому, що інтерпретація автокореляції як кореляції забезпечує безмасштабний показник сили статистичної залежності, а також тому, що нормалізація впливає на статистичні властивості оцінених автокореляцій.

Для спільно широких смислових стаціонарних стохастичних процесів функція перехресної кореляції має таку властивість симетрії:

$$R_{XY}(t_1, t_2) = \overline{R_{YX}(t_2, t_1)}. \quad (27)$$

Відповідно для спільних стаціонарних стохастичних процесів

$$R_{XY}(\tau) = \overline{R_{YX}(-\tau)}. \quad (28)$$

Перехресні кореляції корисні для визначення затримки в часі між двома сигналами, наприклад, для визначення часових затримок для поширення акустичних сигналів через мікрофонний масив. Після обчислення перехресної кореляції між двома сигналами, максимальний (або мінімальний, якщо сигнали негативно корелюють) функції перехресної кореляції вказує момент часу, коли сигнали найкраще вирівняні; тобто затримка часу між двома сигналами визначається аргументом максимуму перехресної кореляції, як:

$$\tau_{delay} = arg \max_{t \epsilon \mathbb{R}}((f \star g)(t)). \quad (29)$$

Нульова нормалізована перехресна кореляція. Для програм обробки зображень, у яких яскравість зображення та шаблону може змінюватись в залежності від освітлення та експозиції, зображення можна спочатку нормалізувати. Зазвичай це робиться на кожному кроці шляхом віднімання середнього та ділення на стандартне відхилення. Тобто, перехресна кореляція шаблону, $t(x,\ y)$ з частиною головного зображення, $f(x,\ y)$ виглядатиме так:

$$\frac{1}{n}\sum_{x,y}\frac{1}{\sigma_f \sigma_t}(f(x,y) - \mu_f)(t(x,y) - \mu_t). \quad (30)$$

де n – кількість пікселів в , $t(x,\ y)$ і , $f(x,\ y)$

μ_f – середнє значення f і σ_f і називається стандартним відхиленням f

У функціональному аналізі це можна розглядати як крапковий добуток двох нормалізованих векторів. Тобто, якщо

$$F(x,y) = f(x,y) - \mu_f. \quad (31)$$

і

$$T(x,y) = t(x,y) - \mu_t. \quad (32)$$

то зазначена вище сума дорівнює:

$$\left\langle \frac{F}{\|F\|}, \frac{T}{\|T\|} \right\rangle. \quad (33)$$

де $\langle \cdot, \cdot \rangle$ є внутрішнім продуктом і $\|\cdot\|$ є простором Lp.

Таким чином, якщо f і t - справжні матриці, їх нормалізована перехресна кореляція дорівнює косинусу кута між одиничними векторами F і T, таким чином 1 буде тоді і тільки тоді, коли F дорівнює T, помножений на позитивний скаляр.

Нормалізована кореляція - це один із методів, що застосовуються для узгодження шаблонів, процес, який використовується для пошуку частот зразка або об'єкта в зображенні. Це також двовимірна версія коефіцієнта кореляції Пірсона.

Нормалізована перехресна кореляція схожа на нульову нормалізовану перехресну кореляцію з тією лише різницею, що не потрібно віднімати середнє локальне значення інтенсивності:

$$\frac{1}{n} \sum_{x,y} \frac{1}{\sigma_f \sigma_t} f(x,y)(t(x,y). \quad (34)$$

Необхідно бути обережними при використанні поперечної кореляції для нелінійних систем. За певних обставин, які залежать від властивостей вхідних

даних, перехресна кореляція між входом і виходом системи з нелінійною динамікою може бути абсолютно сліпою до певних нелінійних ефектів. Ця проблема виникає через те, що деякі квадратичні моменти можуть дорівнювати нулю, і можна неправильно припустити, що між двома сигналами існує невелика "кореляція" (у значенні статистичної залежності), коли насправді два сигнали сильно пов'язані з нелінійною динамікою.

3. АЛГОРИТМИ ОБРОБКИ ЗОБРАЖЕННЯ

3.1 Опис алгоритмів

Алгоритм обробки зображення лягає в основі програмного забезпечення, котре поставлено за ціль реалізувати. Потрібно було протестувати, проаналізувати отримані результати, порівняти з теоретичними відомостями, знайти найефективніший алгоритм. Зображення, котре використовували для аналізу – рис.3.1:

Рисунок 3.1 – зображення для аналізу роботи алгоритмів

З цього зображення взяли область в центрі екрану для збільшення швидкодії дії аналізу та якісної роботи програми. Параметри області – 525 на 450 пікселів.

3.1.1 Метод TM_SQDIFF

Знаходить квадрат різниці. Найкращим варіантом буде 0, а негативні результати будуть більше 0. Формула знаходження:

$$R(x,y) = \sum_{x',y'}(T(x',y') - I\,(x + x', y + y'))^2, \quad (35)$$

де I – вхідне зображення;

T – шаблон зображення;

R – результат порівняння;

x' - приймає значення від 0 до w – 1;

y' - приймає значення від 0 до h – 1;

При заданому фіксуванні локального мінімуму було отримано успішний результат у точці [352, 170] рис.3.2-3.3. Затрачений час – 65 тактів процесора.

```
Mediocre work time  - 65
Min and max Value: -1.28475e-10,    1
Local minimum/maximum location:    352,   170
```

Рисунок 3.2 – отримані дані для аналізу SQDIFF при loc min

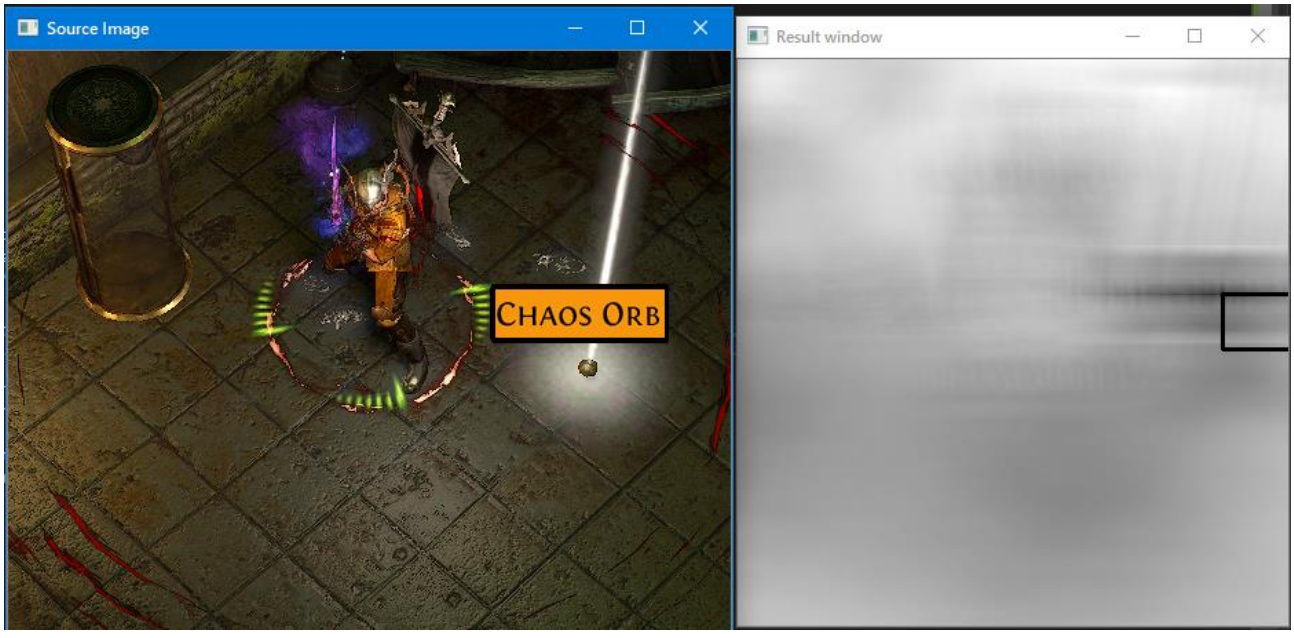

Рисунок 3.3 – результат після використання SQDIFF при loc min

При заданні орієнтування на максимальне значення, алгоритм дає збій рис.3.4-3.5. Затрачений час - 66 тактів процесора.

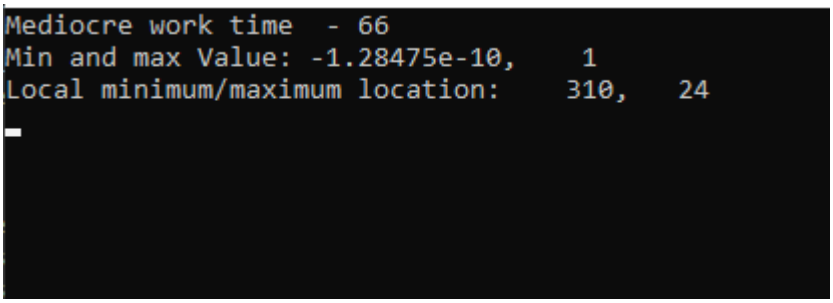

Рисунок 3.4 – отримані дані для аналізу SQDIFF при loc max

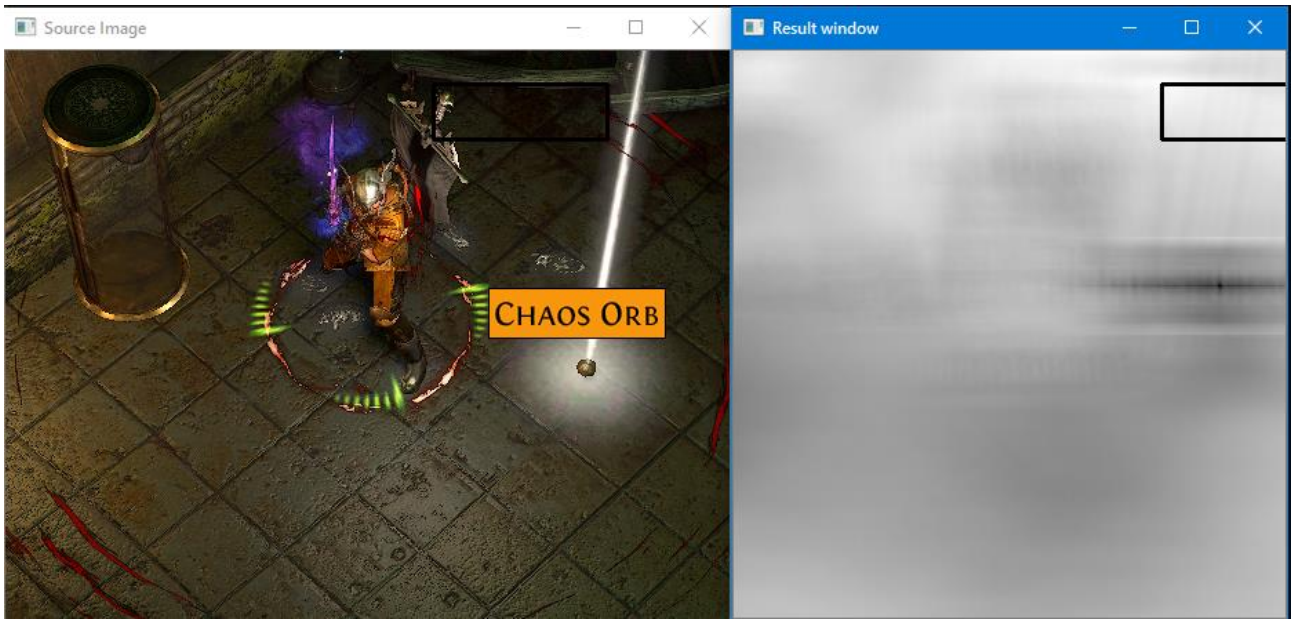

Рисунок 3.5 – результат після використання SQDIFF при loc max

3.1.2 Метод TM_SQDIFF_NORMED

Знаходить квадрат різниці, а потім нормалізує отриманий результат. Найкращим варіантом для повернення буде 0. Формула:

$$R(x,y) = \frac{\sum_{x',y'}(T(x',y') - I\,(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T(x',y')^2 * I\,(x+x',y+y')^2}}, \quad (34)$$

де I – вхідне зображення;

T – шаблон зображення;

R – результат порівняння;

x' - приймає значення від 0 до w – 1;

y' - приймає значення від 0 до h – 1;

Координати отримані при фіксуванні локального мінімуму: [352, 170] рис3.6-3.7. В даному випадку алгоритм справляється з задачею. Затрачений час - 74 такти процесора.

```
Mediocre work time  - 74
Min and max Value: 1.06648e-10,    1
Local minimum/maximum location:    352,   170
```

Рисунок 3.6 – отримані дані для аналізу SQDIFF_NORMED при loc min

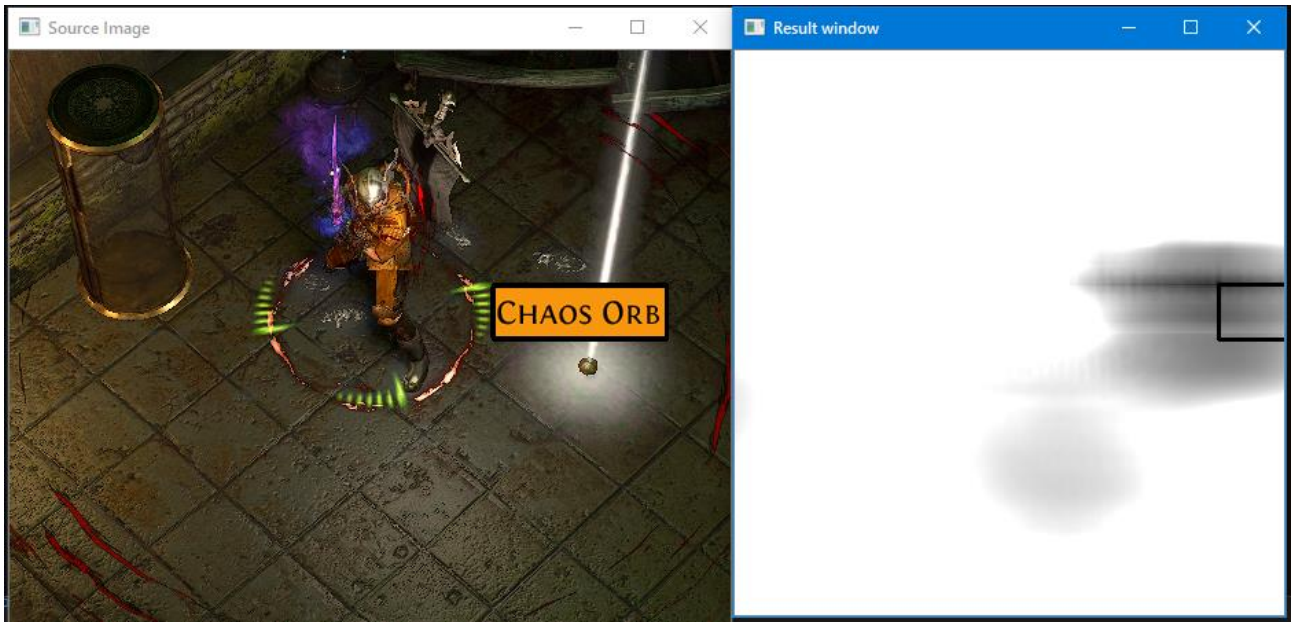

Рисунок 3.7 – результат після використання SQDIFF_NORMED при loc min

При пошуку за локальним максимум у масиві, алгоритм не справляється з задачею, котру поставили перед ним рис.3.8-3.9. Координати повернені після роботи: [0, 0]. Вкрай не вірний результат. Затрачений час - 74 такти процесора.

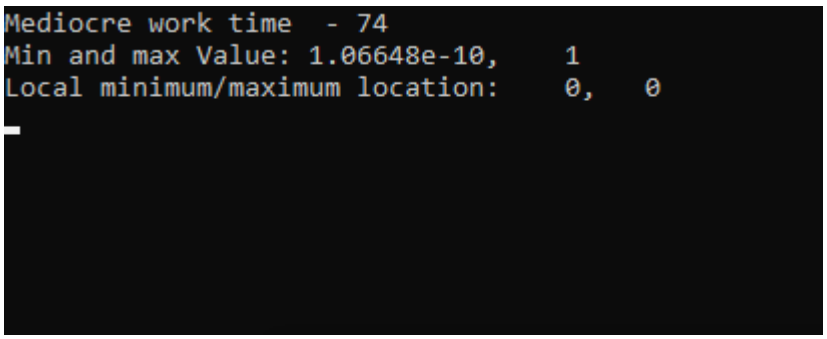

Рисунок 3.8 – отримані дані для аналізу SQDIFF_NORMED при loc max

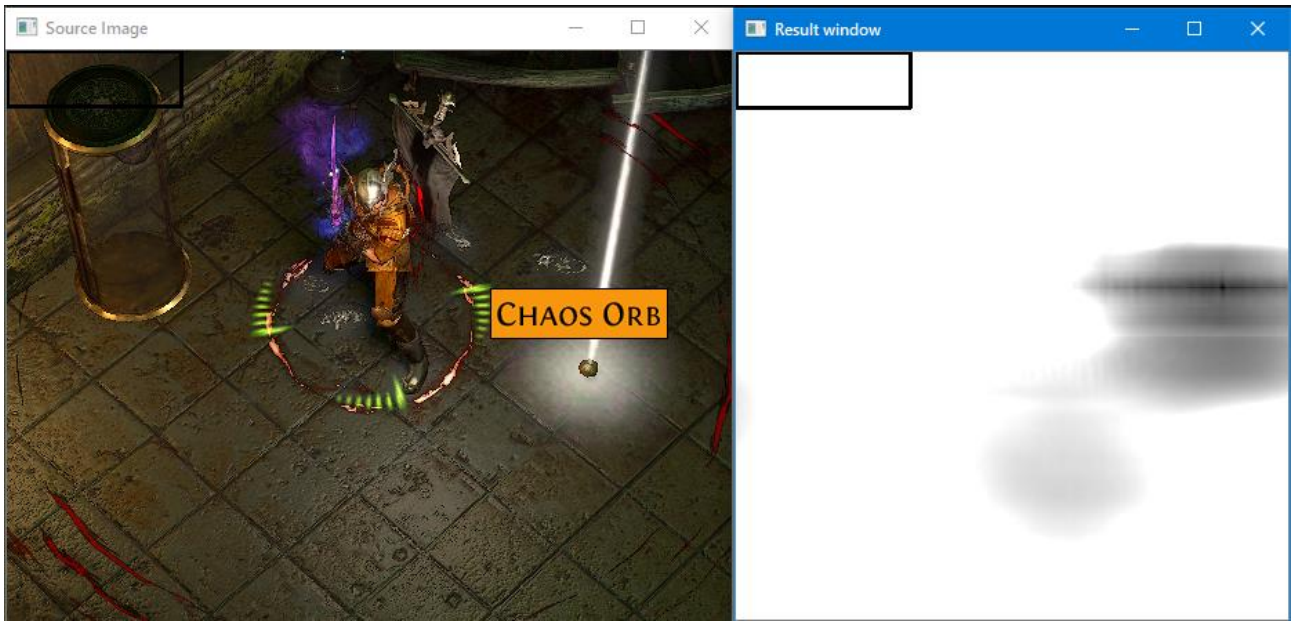

Рисунок 3.9 – результат після використання SQDIFF_NORMED при loc max

3.1.3 TM_CCORR

Кореляційний метод співпадіння. Мультиплікативно фіксує шаблон на вхідному зображенні. Найкращим для нього результатом буде велике значення, а 0 чи близьке до нього буде найгіршим. Формула:

$$R(x,y) = \sum_{x',y'}(T(x',y') * I\,(x + x', y + y'))\,. \quad (36)$$

де I – вхідне зображення;

T – шаблон зображення;

R – результат порівняння;

x' - приймає значення від 0 до w – 1;

y' - приймає значення від 0 до h – 1;

При пошуку локального мінімуму результат повинен бути невірним рис.3.10-3.11. Затрачений час - 55 тактів процесора.

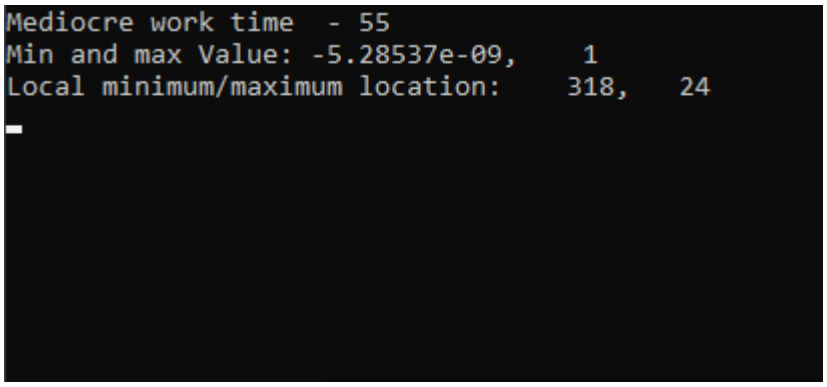

Рисунок 3.10 – отримані дані для аналізу CCORR при loc min

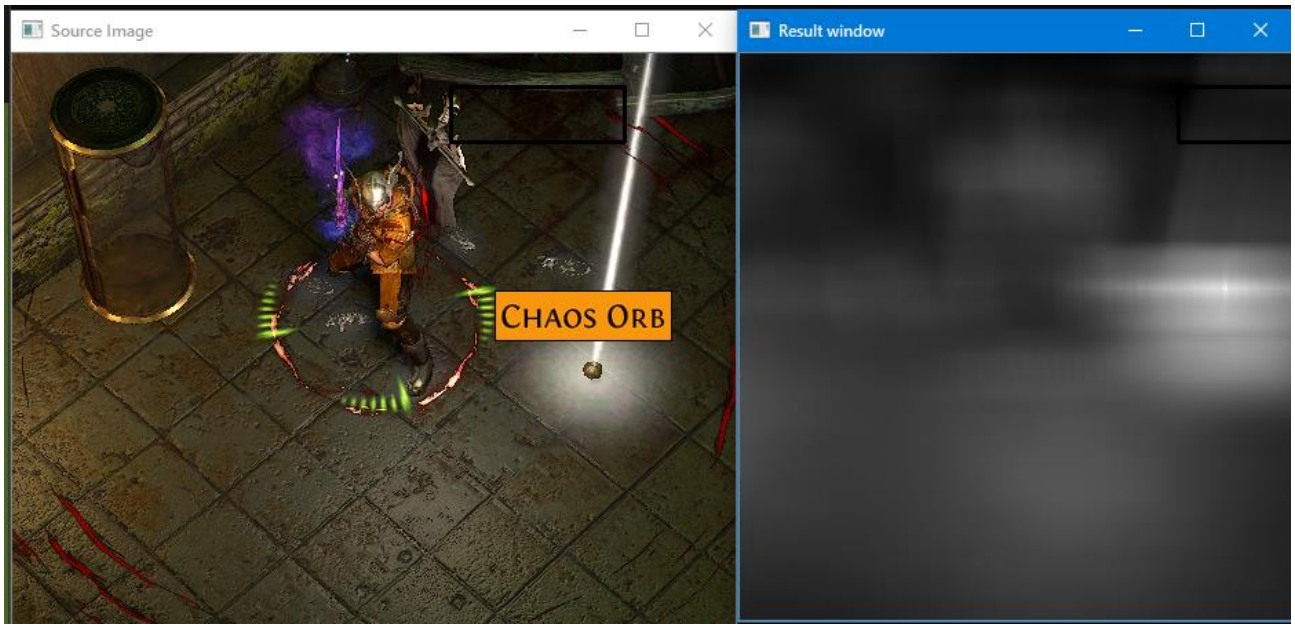

Рисунок 3.11 – результат після використання CCORR при loc min

Отриманий результат підтверджує очікування. Отримані значення, мінімум та максимум відповідно: -5.28537e-09 і 1, координати [318, 24].

При пошуку локального максимум, алгоритм показує вірний результат рис.3.12-3.13. Координати [352, 170]. Затрачений час - 55 тактів процесора.

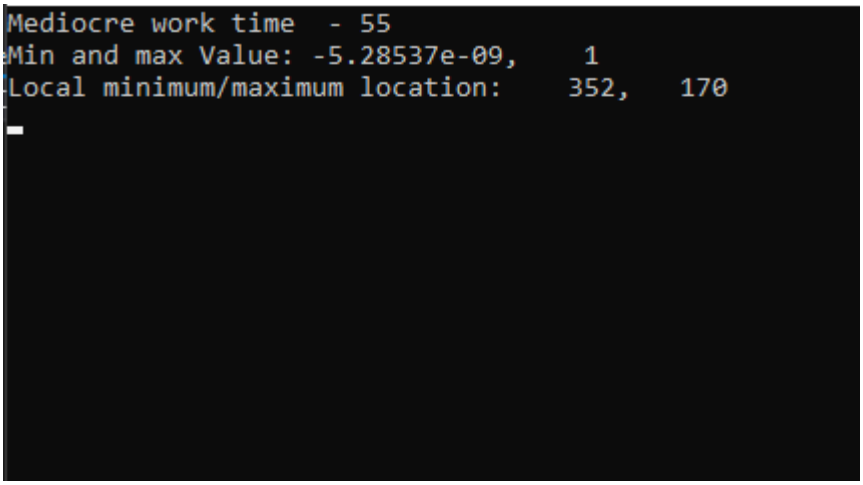

Рисунок 3.12 – отримані дані для аналізу CCORR при loc max

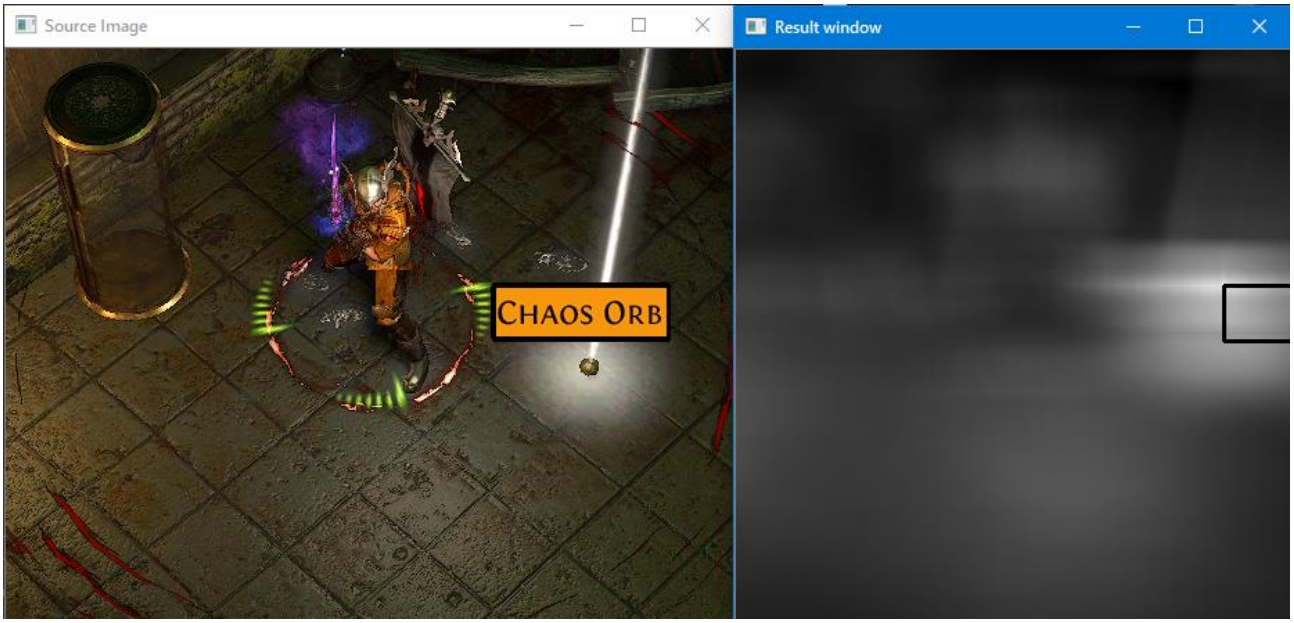

Рисунок 3.13 – результат після використання CCORR при loc max

3.1.4 TM_CCORR_NORMED

Нормалізований метод узгодження крос-кореляції. Як і для свого попередника, даний алгоритм поверне 0 при не співпадінні та більше нуля при знаходженні. Формула:

$$R(x,y) = \frac{\sum_{x',y'} T(x',y') * I\,(x+x', y+y')}{\sqrt{\sum_{x',y'} T(x',y')^2 * I\,(x+x', y+y')^2}}, \quad (37)$$

де I – вхідне зображення;

T – шаблон зображення;

R – результат порівняння;

x' - приймає значення від 0 до w – 1;

y' - приймає значення від 0 до h – 1;

Очікуваним результатом буде: при пошуку максимального результат задовільний рис.3.14-3.15, при локальному мінімумі результат незадовільний рис.3.16-3.17. Затрачений час - 74 такти процесора.

```
Mediocre work time  - 74
Min and max Value: 1.63995e-08,    1
Local minimum/maximum location:    352,   170
```

Рисунок 3.14 – отримані дані для аналізу CCORR_ NORMED при loc max

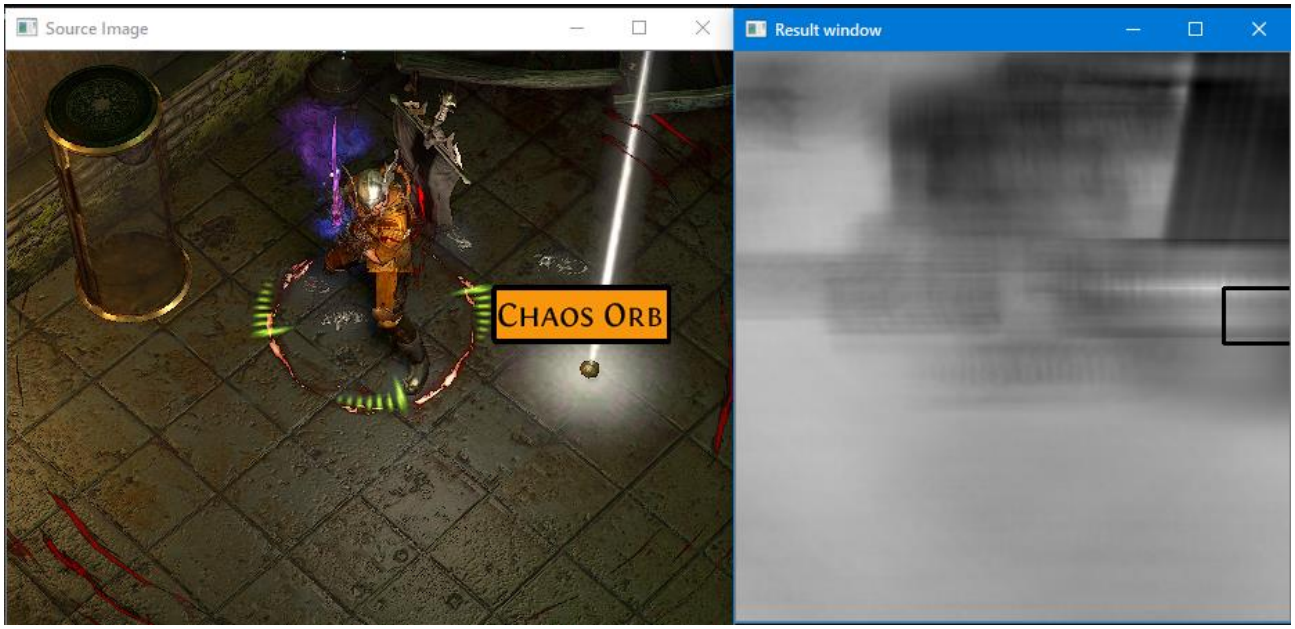

Рисунок 3.15 – результат після використання CCORR_ NORMED при loc max

Отримані результати: значення - 1.63995e-08 та 1, локальний максимум розташований у точці з координатами - [352, 170]. Отримали такий результат після пошуку локального мінімуму – [335, 24]. Затрачений час - 75 тактів процесора.

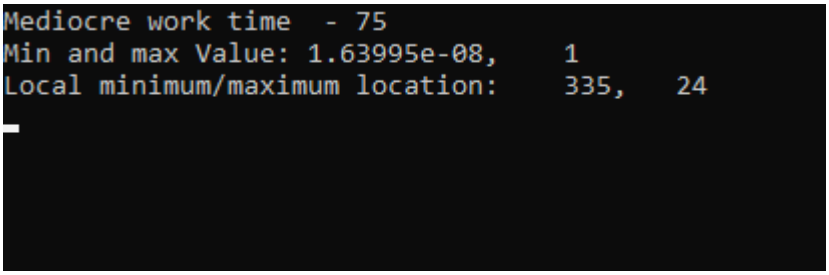

Рисунок 3.16 – отримані дані для аналізу CCORR_ NORMED при loc min

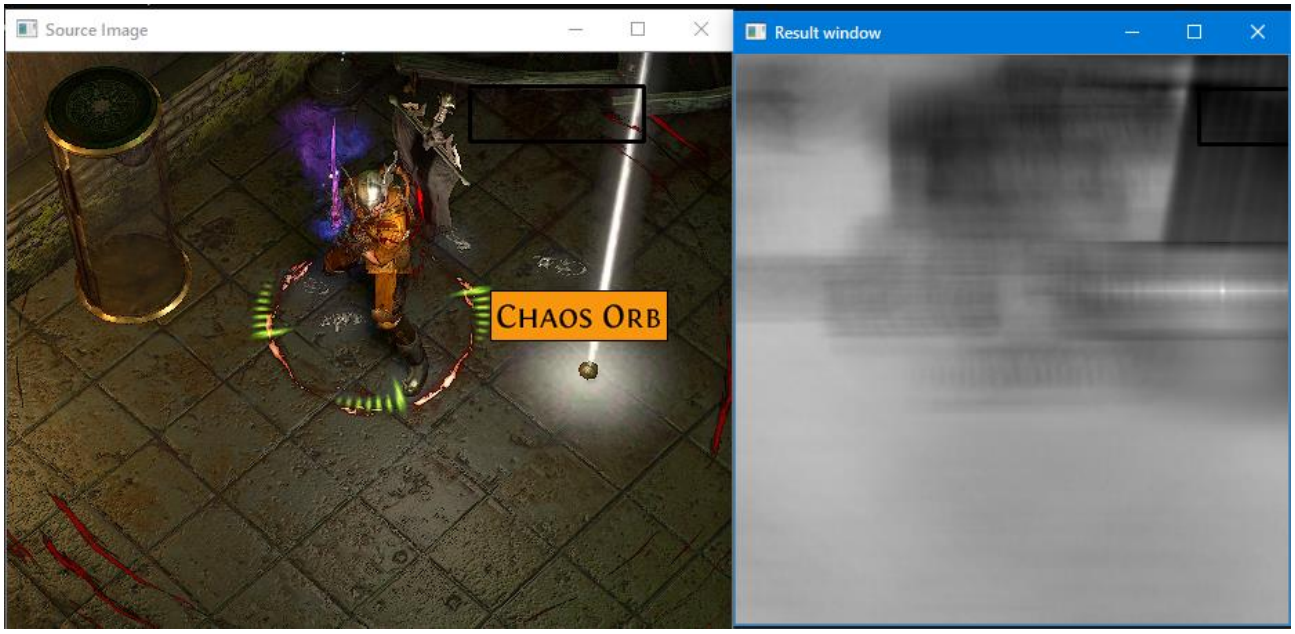

Рисунок 3.17 – результат після використання CCORR_ NORMED при loc min

3.1.5 TM_CCOEFF

Метод узгодження коефіцієнта кореляції.

$$R(x,y) = \sum_{x',y'} T'(x',y') * I'\ (x + x', y + y');\quad (38)$$

$$T'(x',y') = T(x',y') - \frac{1}{w*h} * \sum_{x'',y''} T(x'',y'');\quad (39)$$

$$I'\ (x + x', y + y') = I\ (x + x', y + y') - \frac{1}{w*h} * \sum_{x'',y''} I\ (x + x'', y + y''),\quad (40)$$

де I – вхідне зображення;

T – шаблон зображення;

R – результат порівняння;

x' - приймає значення від 0 до w – 1;

y' - приймає значення від 0 до h – 1;

w – ширина шаблону зображення;

h – висота шаблону зображення;

Цей метод узгоджує шаблон відносно його середнього значення до зображення відносно його середнього зображення. Для заданого алгоритму найкращим співпадінням буде 1, а провал буде з значенням -1, 0 показуватиме, що кореляції немає. При пошуку локального мінімуму отримуємо точку з координатами [349, 179] та затрачений час - 102 такти процесора рис.3.18-3.19.

```
Mediocre work time  - 102
Min and max Value: -5.39493e-09,    1
Local minimum/maximum location:    349,   179
```

Рисунок 3.18 – отримані дані для аналізу CCOEFF при loc min

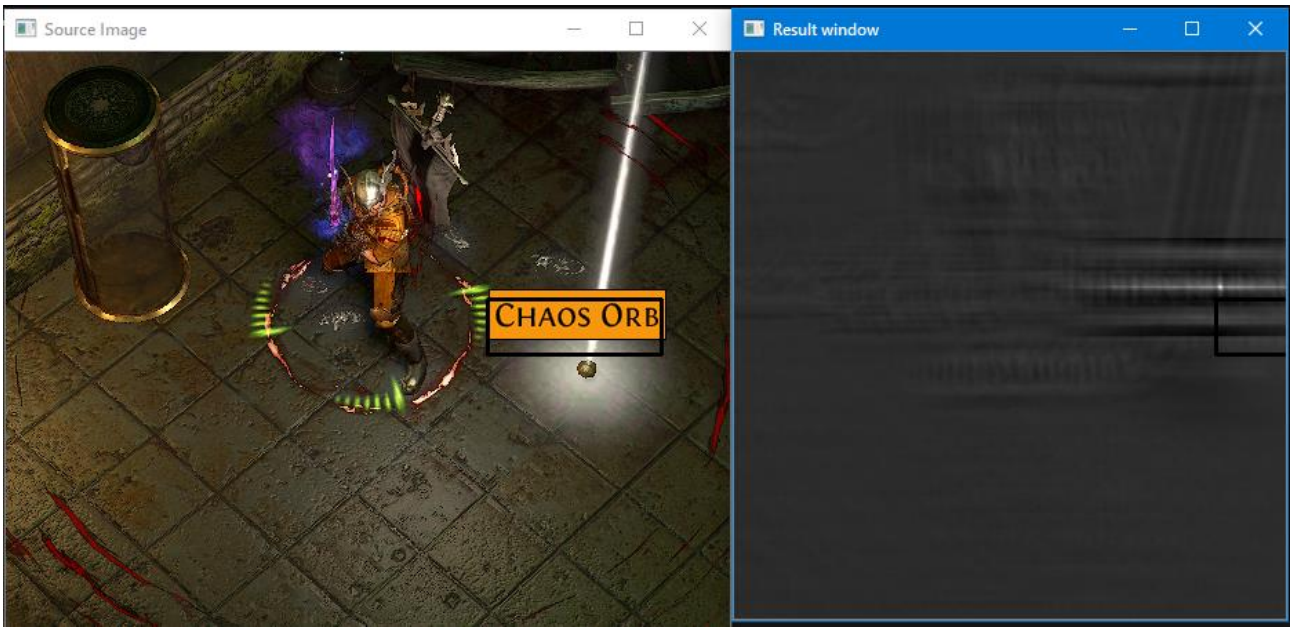

Рисунок 3.19 – результат після використання CCOEFF при loc min

При пошуку локального максимуму, одержуємо точку з координатами [352, 170] та затрачений час - 103 такти процесора рис.3.20-3.21.

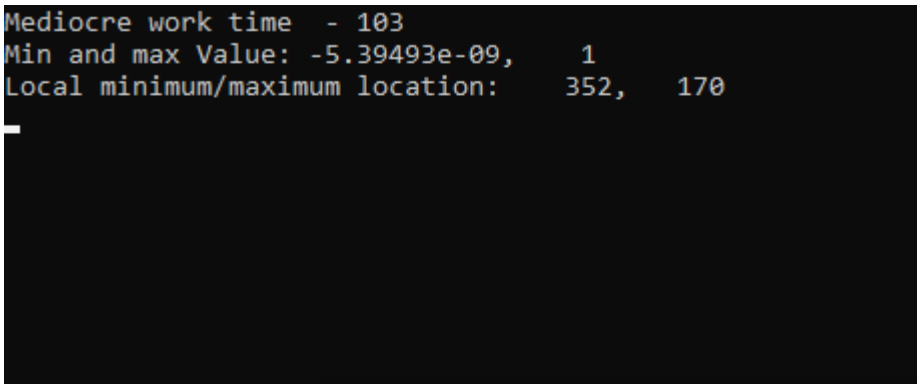

Рисунок 3.20 – отримані дані для аналізу CCOEFF при loc max

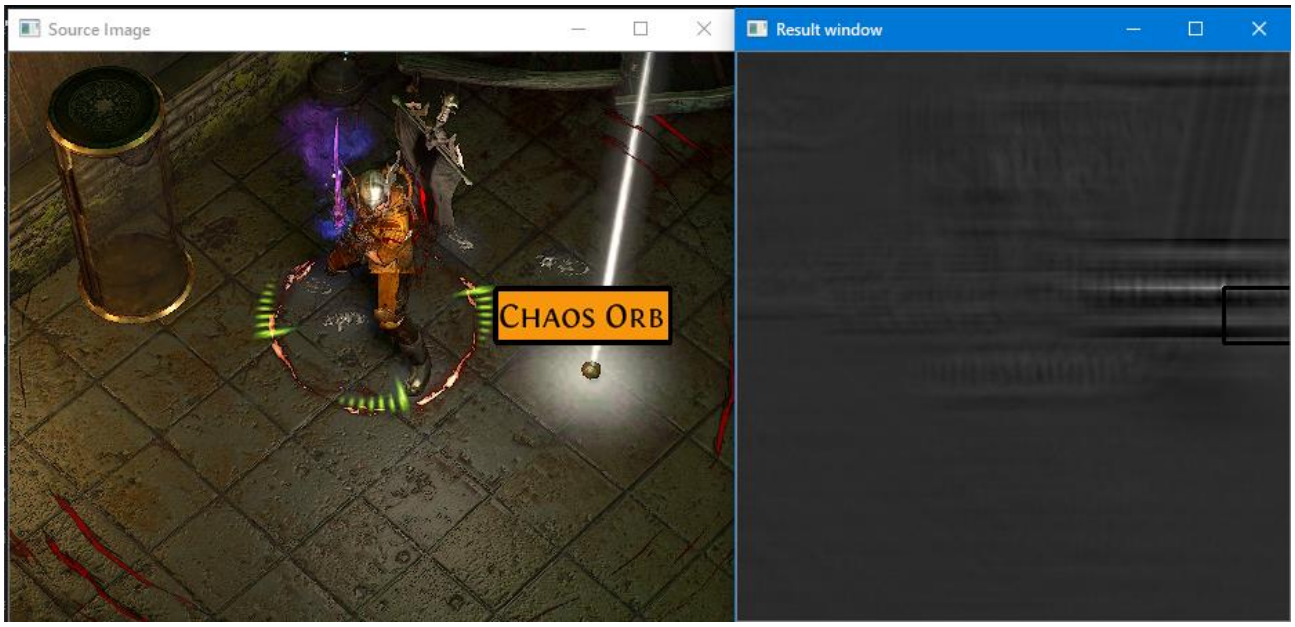

Рисунок 3.21 – результат після використання CCOEFF при loc max

3.1.6 TM_ CCOEFF_NORMED

При відносному співпадінні буде повертати позитивне значення, а в іншому разі – негативне.

$$R(x,y) = \frac{\sum_{x',y'} T'(x',y') * I'(x+x',y+y')}{\sqrt{\sum_{x',y'} T'(x',y')^2 * I'(x+x',y+y')^2}}, \quad (41)$$

$$T'(x',y') = T(x',y') - \frac{1}{w*h} * \sum_{x'',y''} T(x'',y''), \quad (42)$$

$$I'(x + x', y + y') = I(x + x', y + y') - \frac{1}{w*h} * \sum_{x'',y''} I(x + x'', y + y''), \quad (43)$$

де I – вхідне зображення;

T – шаблон зображення;

R – результат порівняння;

x' - приймає значення від 0 до w – 1;

y' - приймає значення від 0 до h – 1;

w – ширина шаблону зображення;

h – висота шаблону зображення;

Пошук локального мінімуму показав такі значення: -1.58317e-09 та 1, мінімальне та максимальне відповідно. Знайдена точка розташована за координатами – [351, 135] рис.3.22-3.23. Результат очевидний – неправильне визначення. Затрачений час - 87 тактів процесора.

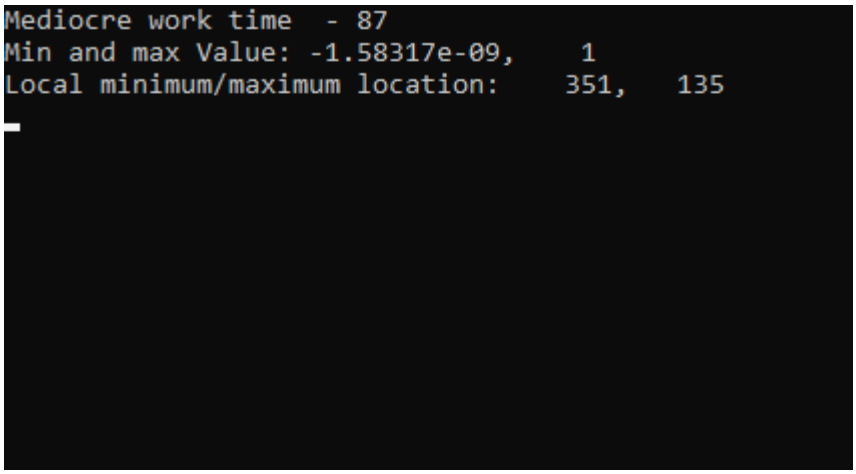

Рисунок 3.22 – результат після використання CCOEFF_NORMED при loc min

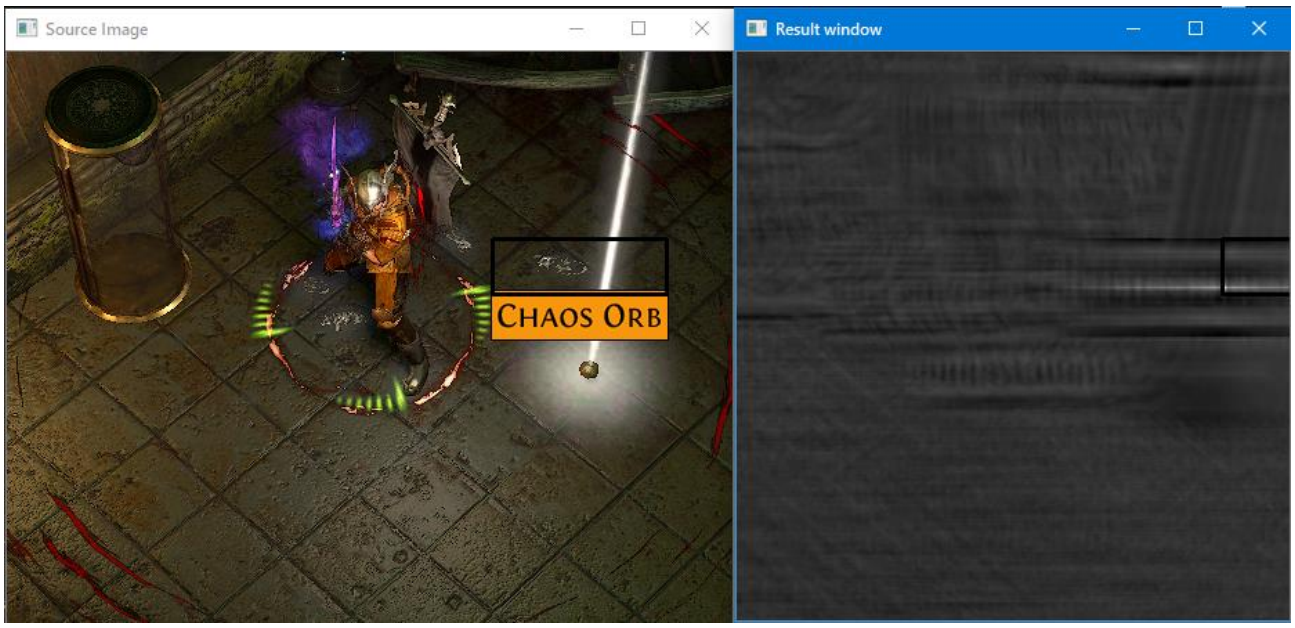

Рисунок 3.23 – результат після використання CCOEFF_NORMED при loc min

Підставимо у пошук значення локального максимуму. Знайдена точка розташована за координатами – [352, 170]. Результат співпадає з очікуваннями. Затрачений час - 85 тактів процесора рис.3.24-3.25.

```
Mediocre work time  - 85
Min and max Value: -1.58317e-09,    1
Local minimum/maximum location:    352,   170
```

Рисунок 3.24 – результат після використання CCOEFF_NORMED при loc max

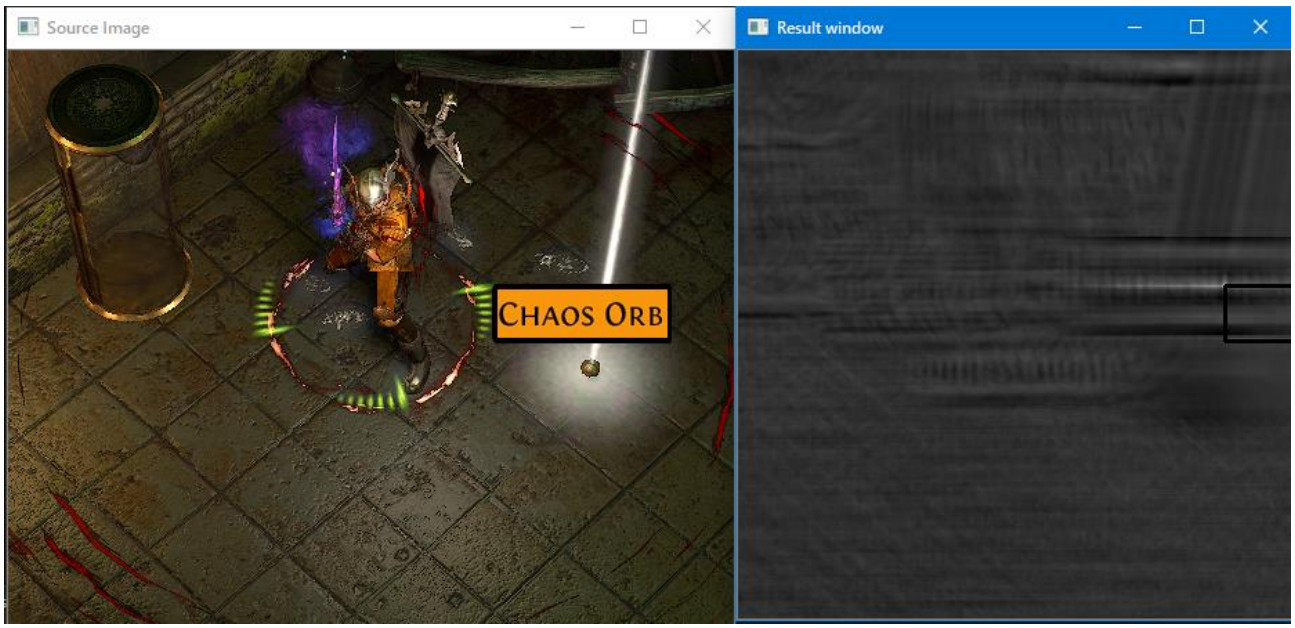

Рисунок 3.25 – результат після використання CCOEFF_NORMED при loc max

3.2 Ресурсні затрати алгоритмів

Розглянемо нижче затрати оперативної пам’яті при роботі алгоритмів. В середньому процесор загружений на 25% у всіх алгоритмів рис.3.26-3.31.

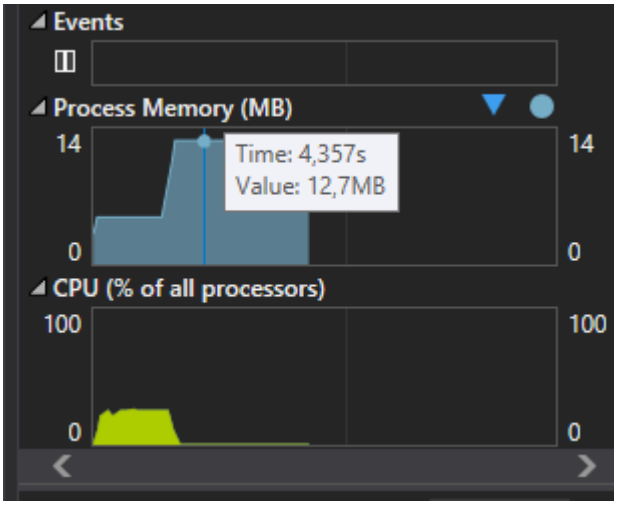

Рисунок 3.26 – затрати на SQDIFF

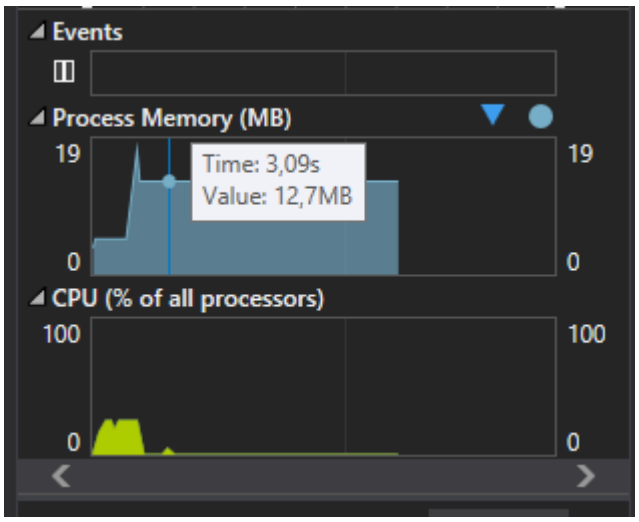

Рисунок 3.27 – затрати на SQDIFF_NORMED

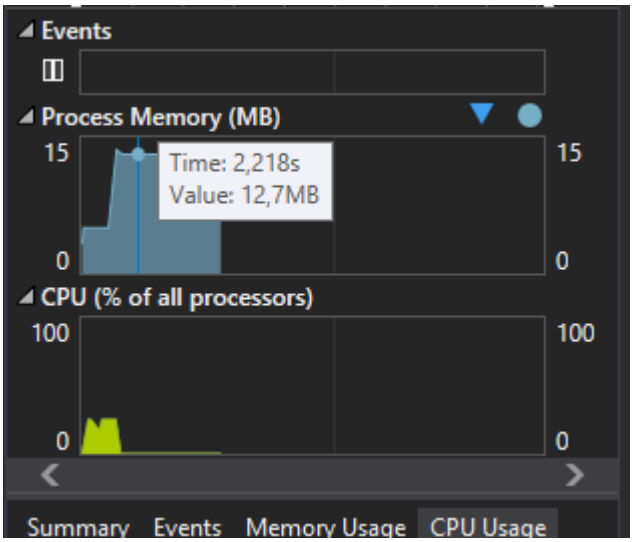

Рисунок 3.28 – затрати на CCORR

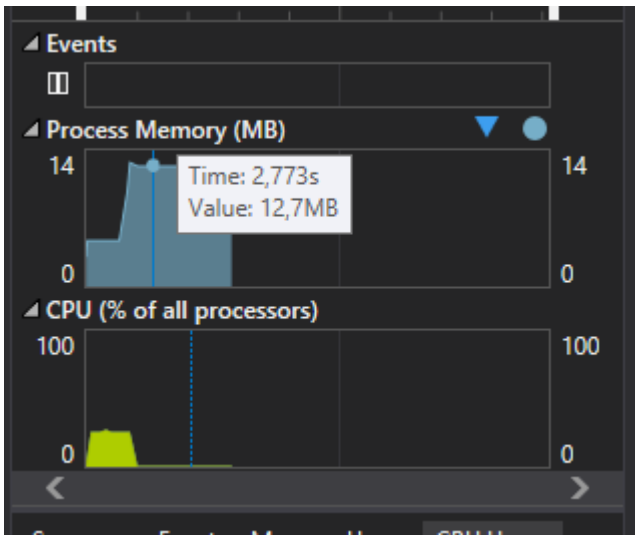

Рисунок 3.29 – затрати на CCORR_NORMED

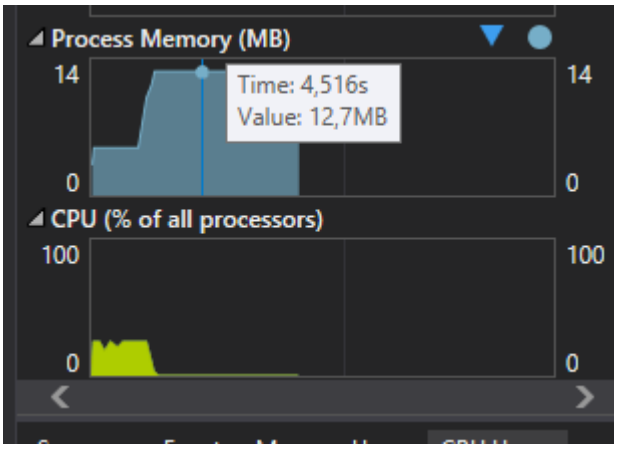

Рисунок 3.30 – затрати на CCOEFF

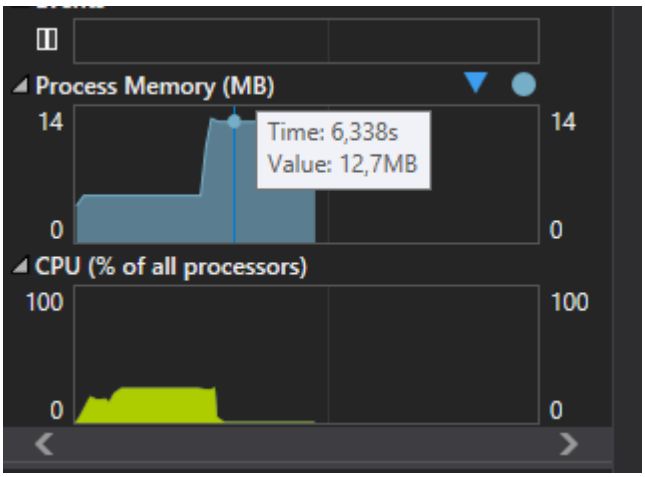

Рисунок 3.31 – затрати на CCOEFF_NORMED

З результатів можна зробити висновок, що всі алгоритми використовують однакову кількість оперативної пам’яті та однакову роблять нагрузку на процесор комп’ютера. Отже, опір при виборі алгоритму будемо брати на швидкодію та правильність виконання.

3.3 Аналіз роботи алгоритмів

Алгоритми SQDIFF і SQDIFF_NORMED показують найкраще співпадіння для результатів на котрих переважають темні відтінки кольорів, з іншого боку для алгоритмів CCOR, CCOR_NORMED, CCOEFF, CCOEFF_NORMED чим яскравіша область тим краще.

Для знаходження часу виконання алгоритмів було використано певний алгоритм знаходження середнього часу виконання програми пошуку. Заміри проводилися 28 раз для кожного алгоритму по черзі. Перших два виміри відкидаються. Серед елементів часу, котрі залишилися знаходимо 3 максимальних та 3 мінімальних, які ми також відкидаємо. Далі знаходимо мінімальний та максимальний елементи у масиві часу і встановлюємо їм значення -1; це потрібно повторити 3 рази. Таким чином, у суму всіх вимірів попадуть значення усі крім: перших двох, трьох найбільших та трьох найменших. Такою вибіркою можна викинути із обігу випадкові перепади у роботі процесора чи пам’яті. На виході із 28 результатів отримуємо 20. Отримані результати записано у табл.3.1.

Таблиця 3.1 – результати часових вимірів

Назва алгоритму	Результат часових вимірів (в тактах)	
	для мінімуму	для максимуму
SQDIFF	65	66
SQDIFF_NORMED	74	74
CCORR	55	55
CCORR_NORMED	74	75
CCOEFF	102	103
CCOEFF_NORMED	87	85

З таблиці видно, що різниці між пошуками локальних максимумів чи мінімумів по часовому пріоритеті немає. А, якщо і є, то різниця у 1 процесорний такт не є суттєвою, тому нею можна нехтувати. Отже, алгоритми працюють однаково ефективно у цьому плані, а різниця з'являється лише у правильності результатів. Якщо вибрати алгоритми, котрі справляються з задачою пошуку, отримуємо такий список:

- SQDIFF при мінімальному значенні;
- SQDIFF_NORMED при мінімальному значенні;
- CCORR при максимальному значенні;
- CCORR_NORMED при мінімальному значенні;
- CCOEFF при максимальному значенні;
- CCOEFF_NORMED при максимальному значенні.

3.4 Висновок про аналізовані алгоритми

Після проведення дослідів на швидкодію, ресурсозатратність та правильність виконання поставленої задачі, робимо висновок, що найкращим алгоритмом за критерієм швидкодії для пошуку співпадінь за певним шаблоном в нашому випадку буде SQDIFF, за критерієм найбільш точного розпізнавання CCOEFF_NORMED. Але, оскільки нормовані версії мають більшу точність результатів, алгоритм SQDIFF – відкинемо, а нормована версія даного алгоритму - SQDIFF_NORMED вимагає стільки часу, скільки і алгоритм CCORR_NORMED, котрий пропонує більшу точність (через свою складність) за такий же час. В порівнянні алгоритмів CCORR_NORMED і CCOEFF_NORMED, отримуємо виграш у часі на 13% для першого. Звісно, дані показники актуальні лише для комп'ютерної системи, на котрій виконувались тести, оскільки швидкодія буде залежати від складу комп'ютерної системи, її загруженості і доступності ресурсів комп'ютера. Отже, із двох алгоритмів виберемо той, який швидший - CCOEFF_NORMED.

ВИСНОВКИ

В процесі виконання дипломного проєкту було розроблено програмне забезпечення на базі комп’ютерного зору для автоматизації ігрового процесу користувача. Під час роботи було реалізовано алгоритм для створення, аналізу та використання зображення задля виконання поставленої задачі. Даний алгоритм показує високі показники швидкодії та працездібності, що задовільняє поставлені перед ним потреби. Розроблений програмний продукт забезпечує вказаний функціонал за допомогою комп’ютерного зору, який є не наявним у аналогічного стороннього програмного забезпечення.

Дане програмне забезпечення показує велику еластичність. Наприклад, для аналогічних задач без зміни алгоритму, а лише з заданням нового шаблону, можна досягти такого ж результату, як і для обраної гри. Також, можна окремо використовувати лише модуль збереження зображень.

Також, потрібно зазначити, що можливо розширити у майбутньому функціонал розробленої програми. Векторами можливої розробки можуть бути: автоматичне сортування предметів, автоматичний продаж, створення певного виду ігрової одежі з вказаними параметрами, автовідповідач на повідомлення від інших гравців, проведення автоматичної торгівлі, без втручання користувача, з іншими гравцями.

ЛІТЕРАТУРА

1. Joshua Glazer, Sanjay Madhav Multiplayer game programming: 2016. 386 pages.
2. Marr, David Vision: a computational investigation into the human representation and processing of visual information: 1982. 422 pages.
3. Horn, Berthold «Robot vision». – 1986. 538 pages.
4. Ballard, Dana H. (Dana Harry) «Computer vision». – 1982. 558 pages.
5. Richard Szeliski «Computer Vision: Algorithms and Applications». - 2010. 812 pages.
6. Pedram Azad, Tilo Gockel, Rüdiger Dillmann «Computer Vision: Principles and Practice». – 2008. 320 pages.